Bar Recursion over Finite Partial Functions

Paulo Oliva[1] and Thomas Powell*[2]

[1]Queen Mary University of London
[2]Technische Universität Darmstadt
[1]p.oliva@qmul.ac.uk
[2]powell@mathematik.tu-darmstadt.de

December 14, 2016

Abstract

We introduce a new, demand-driven variant of Spector's bar recursion in the spirit of the Berardi-Bezem-Coquand functional of [4]. The recursion takes place over finite partial functions u, where the control parameter ω, used in Spector's bar recursion to terminate the computation at sequences s satisfying $\omega(\hat{s}) < |s|$, now acts as a guide for deciding exactly where to make bar recursive updates, terminating the computation whenever $\omega(\hat{u}) \in \mathrm{dom}(u)$. We begin by exploring theoretical aspects of this new form of recursion, then in the main part of the paper we show that demand-driven bar recursion can be directly used to give an alternative functional interpretation of classical countable choice. We provide a short case study as an illustration, in which we extract a new bar recursive program from the proof that there is no injection from $\mathbb{N} \to \mathbb{N}$ to $\mathbb{N}$, and compare this with the program that would be obtained using Spector's original variant. We conclude by formally establishing that our new bar recursor is primitive recursively equivalent to the original Spector bar recursion, and thus defines the same class of functionals when added to Gödel's system T.

Keywords. Spector's bar recursion, Dialectica interpretation, program extraction, countable choice.

MSC 2010. 03D65, 03F03, 03F10, 03F25.

1 Introduction

In 1962 C. Spector extended Gödel's functional, or *Dialectica*, interpretation of classical arithmetic [13] to full classical analysis by proving that the functional interpretation of the negative translation of countable choice, and hence full arithmetical comprehension, could be realized by a novel form of recursion which has come to be known as *Spector's bar recursion* [24]. Since then, this seminal work has been extended in several ways, and in particular a number of novel variants of bar recursion have been devised to give computational interpretations to classical analysis in new settings, to the extent that bar recursion, in one form or another, is one of the most recognisable methods of giving a computational interpretation to mathematical analysis.

Spector's original aim was to extend Gödel's proof of the relative consistency of Peano arithmetic to classical analysis. For this purpose, bar recursion is very well suited, allowing us to elegantly and easily expand the soundness of the Dialectica interpretation to incorporate the double negation shift and thus classical countable choice. However, in recent decades applications of proof interpretations such as the Dialectica interpretation and modified realizability have moved away from foundational concerns and towards the more practical issue of extracting computational content from proofs. In line with this shift of emphasis comes an increasing interest in how the modes of computation assigned to non-constructive principles behave.

*Corresponding author: **Email.** powell@mathematik.tu-darmstadt.de, **Phone.** (+49)06151-16-22844, **Address.** Fachbereich Mathematik, Technische Universität Darmstadt, Schlossgartenstrasse 7, Darmstadt, 64289, Germany

From this perspective, it could be argued that traditional bar recursion is not necessarily the best method of interpreting countable choice principles. The defining characteristic of Spector's bar recursion is that it carries out computations over some well-founded tree of finite sequences s, where the value at each node s can only depend on the immediate children $s * x$ of that node. One can think of Spector bar recursion as performing a depth-first search on well-founded trees. But it often happens that the values computed at several nodes of this tree are irrelevant for computing the desired approximation to the choice sequence, and one would like to avoid such unnecessary computations.

The purpose of this paper is to introduce a new, demand-driven alternative to Spector's bar recursion, in which the order of the recursive calls is not fixed but rather directly controlled by its parameters. We first focus on recursion-theoretic issues, in particular giving an intuitive explanation of why our recursor exists in standard continuous models. We then prove that our new form of recursion is capable of realizing the Dialectica interpretation of countable choice, and moreover argue that (from an algorithmic viewpoint) it can be superior to Spector's original bar recursion because the manner in which it constructs approximations to choice sequences is much more sensitive to its environment. We illustrate this with an example in which we extract realizers from the classical proof that there is no injection from $\mathbb{N} \to \mathbb{N}$ to $\mathbb{N}$. In this case the program based on our demand-driven recursion has a much more intuitive behaviour than that based on Spector bar recursion, and significantly outperforms the latter on a small sample of concrete inputs. Finally, we calibrate the computational strength of this new variant of bar recursion relative to Spector's original definition, showing that despite the algorithmic differences in extracted programs, the two forms of bar recursion are in fact primitive recursively interdefinable, and thus our recursor exists in all the usual models of Spector's bar recursion.

Our variant of bar recursion is in some ways similar to the realizer of countable choice proposed by Berardi et al. in [4], now often called the BBC-functional. In both cases the recursion is carried out in a 'symmetric' rather than a fixed sequential manner. However, the BBC-functional belongs to the world of *realizability*, which typically uses much stronger forms of recursion to interpret choice principles than Spector bar recursion (see [8, 12, 20]). Moreover, the BBC-functional itself has a highly complex behaviour, its demand driven execution coming at the expense that each entry in its output is computed via a completely independent recursion. Our bar recursor is very different in this respect, as it retains a 'memory' of what has already been computed, and simply relaxes the order in which the computation occurs. We discuss the BBC-functional in more detail in Section 5.3.

1.1 Preliminaries

The basic formal theory we work over is fully extensional[1] Heyting arithmetic $\mathsf{E\text{-}HA}^\omega$ in all finite types (and its classical variant $\mathsf{E\text{-}PA}^\omega$), whose quantifier-free fragment is Gödel's system T of primitive recursive functionals (see [16, 25] for full details). The finite types $\mathcal{T}$ are typically defined using the following inductive rules

$$\mathcal{T} ::= \mathbb{N} \mid X \times Y \mid X \to Y$$

We expand these basic types with some standard 'definable' types, including the unit $\mathbf{1}$ and Boolean $\mathbb{B} = \{0, 1\}$ types, finite sequence types X^* and co-product types $X + Y$. We consider *partial sequences* $\mathbb{N} \rightharpoonup X$ to be objects of type $\mathbb{N} \to X + \mathbf{1}$, where $\mathbf{1}$ denotes an 'undefined' value.

Finally, we consider one slightly non-standard type: the type $X^\dagger$ of *finite partial functions*, that is partial sequences $\mathbb{N} \rightharpoonup X$ defined at only finitely many points. This type can be easily simulated as in [5] by $(\mathbb{N} \times X)^*$, in which the partial function that takes values $x_0, \ldots, x_{k-1}$ at arguments $n_0, \ldots, n_{k-1}$ is encoded as the sequence $\langle (n_0, x_0), \ldots, (n_{k-1}, x_{k-1}) \rangle$, although to minimise syntax we treat $X^\dagger$ as primitive and avoid any further details of how it can be precisely encoded using the usual types.

Relative to a suitable encoding for $X^\dagger$ there exists a computable functional $\mathrm{dom}\colon X^\dagger \to \mathbb{N}^*$ which for each finite partial function u encodes its finite domain as a sequence of natural numbers (if u is encoded in the type $(\mathbb{N} \times X)^*$ as described above this functional would simply be the first projection of the sequence, quotiented by equality). In general, for both partial and finite partial sequences u we write $\mathrm{dom}(u) \subseteq \mathbb{N}$ to denote the domain of u, and write $n \in \mathrm{dom}(u)$ whenever u is defined at n. We can assume that membership of $\mathrm{dom}(u)$ is a decidable predicate (i.e. recursive in u).

[1] As it is well-known, the Dialectica interpretation does not validate full extensionality. The system we are describing here, however, is the one we use to *verify* the bar recursive interpretation of countable choice, and also for our inter-definability results.

We write $x\colon X$ or x^X to signify "x is an object of type X", and sometimes write Y^X for the type $X \to Y$. In addition to the basic constructors for dealing with the finite types, $\mathsf{E\text{-}HA}^\omega$ contains variables and quantifiers for all types, the predicates $=_{\mathbb{B}}$ and $=_{\mathbb{N}}$ and corresponding axioms for equality over base types, induction over arbitrary formulas, combinators which allow us to carry out λ-abstraction, and primitive recursors R_X for each type which satisfy

$$\begin{aligned}\mathsf{R}_X^{y,z}(0) &=_X y\\ \mathsf{R}_X^{y,z}(n+1) &=_X z_n(\mathsf{R}_X^{y,z}(n)).\end{aligned}$$

In particular, the recursor of type $\mathbb{N}$ allows us to carry out definition by cases, and also assign characteristic functions to all quantifier-free formulas of $\mathsf{E\text{-}HA}^\omega$. Note that we choose to write $\mathsf{R}_X^{y,z}(n)$ instead of the more common $\mathsf{R}_X(y,z,n)$. This is simply a notational separation between the parameters that remain fixed throughout the recursion, namely y and z, and the parameter over which the recursion takes place, namely n. We follow a similar convention when defining bar recursion schemes below.

Finally, higher-type equality $=_X$ for arbitrary X is defined inductively in terms of $=_{\mathbb{N}}$, and is treated as fully extensional via the axioms

$$\forall f^{X\to Y}, x^X, y^X (x =_X y \to f(x) =_Y f(y)).$$

We will often require extensions of $\mathsf{E\text{-}HA}^\omega/\mathsf{E\text{-}PA}^\omega$ with various principles, notably both the axioms $\mathsf{AC}_{\mathbb{N},X}$ of countable choice and DC of countable dependent choice, which are defined by

$$\mathsf{AC}_{\mathbb{N},X} :\ \forall n^{\mathbb{N}} \exists x^X A_n(x) \to \exists f^{\mathbb{N}\to X} \forall n A_n(f(n))$$

and

$$\mathsf{DC}_X :\ \forall n^{\mathbb{N}}, x^X \exists y^X A_n(x,y) \to \exists f^{\mathbb{N}\to X} \forall n A_n(f(n), f(n+1))$$

respectively, where A is an arbitrary formula in each case. We denote by $\mathsf{AC}_{\mathbb{N}}$ and DC the general axiom schemata ($\mathsf{AC}_{\mathbb{N},X}$) and ($\mathsf{DC}_X$) where X ranges over arbitrary finite types.

1.2 Notation

We make use of the following notational conventions:

- *Finite sequence constructors.* For $s\colon X^*$, $|s|$ denotes the length of s. We use $\langle x_0, \ldots, x_{n-1}\rangle\colon X^*$ to denote the finite sequence constructor. Hence, we write $\langle\rangle$ for the empty sequence and $\langle x\rangle$ for the singleton sequence containing only x.

- *Finite sequence concatenation.* Given two finite sequences $s, t\colon X^*$ we write $s * t$ for their concatenation. For a finite sequence $s\colon X^*$ and an object $x\colon X$ we often write $s * x$ for $s * \langle x\rangle$. We also write $s * \alpha\colon X^{\mathbb{N}}$ to denote the concatenation of the finite sequence $s\colon X^*$ with an infinite sequence $\alpha\colon X^{\mathbb{N}}$.

- *Initial finite sequence.* Given an $\alpha\colon X^{\mathbb{N}}$ we write $[\alpha](n) = \langle\alpha(0), \ldots, \alpha(n-1)\rangle : X^*$ for the finite initial segment of length n of the infinite sequence α.

- *Finite partial function constructors.* We use $\emptyset\colon X^\dagger$ for the finite partial function with empty domain, and $(n, x)\colon X^\dagger$ for the finite partial function defined only at point n with value x.

- *Ordering of finite partial functions.* Given two finite partial functions $u, v\colon X^\dagger$ we write $u \sqsubseteq v$ if the domain of v contains the domain of u; and u and v coincide on the domain of u. If the domain of u is strictly contained in the domain of v we will write $u \sqsubset v$.

- *Merging finite partial functions.* Given two finite partial functions $u, v\colon X^\dagger$ we write $u \oplus v$ to denote the "union" of the two partial functions, where we give priority to the values of u when u and v are both defined at some common point. Given finite sequences $s, t\colon X^*$ we also write $s \oplus t$ since we can think of $s, t\colon X^*$ as finite partial functions of a particular kind.

- *Canonical extensions.* The term $\mathbf{0}_X\colon X$ denotes the some inductively defined "zero object" of each type X, used as a canonical representative of X – we use the convention that $\mathbf{0}_{X^*} = \langle\rangle$ and $\mathbf{0}_{X^\dagger} = \emptyset$. The canonical extension $\hat{s}\colon X^{\mathbb{N}}$ of the finite sequence $s\colon X^*$ is defined by $\hat{s}(i) = s_i$ for $i < |s|$ and else $\hat{s}(i) = \mathbf{0}_X$. The canonical extension $\hat{u}\colon X^{\mathbb{N}}$ of a finite partial function $u\colon X^\dagger$ is defined analogously. Given a function $\omega\colon X^{\mathbb{N}} \to R$ we also talk about its canonical extension, and write $\hat{\omega}$ for the function $\hat{\omega}(s) = \omega(\hat{s})$ so that $\hat{\omega}$ can be either of type $X^* \to R$ or $X^\dagger \to R$. The type of $\hat{\omega}$ will be clear from the context.
- *Partial application.* As a generalisation of currying, given a function $\omega\colon X^{\mathbb{N}} \to R$ and an $s\colon X^*$, we write $\omega_s\colon X^{\mathbb{N}} \to R$ for the function defined by $\omega_s(\alpha) = \omega(s * \alpha)$.
- *Bounded search.* Given any decidable predicate $P(i)$ on $\mathbb{N}$, the term $\mu i \leq n\ .\ P(i)$ returns the least $i \leq n$ satisfying $P(i)$, if it exists, or returns n otherwise.

1.3 Spector's bar recursion

The defining equation of Spector's general bar recursor $\mathsf{BR}_{X,R}$ is given by

$$\mathsf{BR}^{\phi,b,\omega}_{X,R}(s^{X^*}) =_R \begin{cases} b(s) & \text{if } \omega(\hat{s}) < |s| \\ \phi_s(\lambda x^X.\mathsf{BR}^{\phi,b,\omega}_{X,R}(s * x)) & \text{otherwise} \end{cases}$$

where the parameters have type $\phi\colon X^* \to (X \to R) \to R$, $b\colon X^* \to R$ and $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$ and X, R range over arbitrary types. Just as we did with $\mathsf{R}^{y,z}_X(n)$, we also write $\mathsf{BR}^{\phi,b,\omega}_{X,R}(s)$ instead of the more common $\mathsf{BR}_{X,R}(\phi, b, \omega, s)$, so as to highlight the parameter s over which the recursion takes place. In fact, we will often omit the parameters ϕ, b, ω from the superscript of BR when there is no danger of ambiguity.

The parameter ω acts as a 'control' for $\mathsf{BR}^{\phi,b,\omega}_{X,R}(s)$, whose role is to ensure that at some point the recursive calls stop. Therefore Spector's bar recursor is well-founded only if the control parameter eventually satisfies $\hat{\omega}(s * \langle x_0, \ldots, x_{N-1}\rangle) < |s| + N$ (recall that we use the abbreviation $\hat{\omega}(t) = \omega(\hat{t})$) for each sequence of recursive calls s, $s * \langle x_0\rangle$, $s * \langle x_0, x_1\rangle \ldots$. We call this requirement *Spector's condition*, which can be formulated more precisely as

$$\mathsf{Spec}_X\ :\ \forall \omega^{X^{\mathbb{N}} \to \mathbb{N}} \forall \alpha^{X^{\mathbb{N}}} \exists n(\hat{\omega}([\alpha](n)) < n).$$

As demonstrated by Howard using a trick[2] attributed to Kreisel, Spec must be valid in any model of bar recursion.

Proposition 1.1 (Howard/Kreisel [14])**.** $\mathsf{E\text{-}HA}^\omega + (\mathsf{BR}) \vdash \mathsf{Spec}$.

For this reason, BR is not well-defined in the full type structure of all set-theoretic functionals, since Spec is clearly not valid in this structure. However, it is well known to exist in most continuous type-structures (such as the Kleene/Kreisel continuous functionals [15, 17, 25]), and even in the type structure of strongly majorizable functionals [9], which contains non-continuous functionals.

Just as normal primitive recursion forms a computational analogue of induction, bar recursion can be viewed as a computational analogue of the principle of *bar induction*, which is well-known to be (classically) equivalent to dependent choice,

$$\mathsf{BI}\ :\ \forall \alpha^{X^{\mathbb{N}}} \exists n P([\alpha](n)) \wedge \forall t^{X^*}(\forall x^X P(t * x) \to P(t)) \to P(\langle\rangle).$$

Here P is some predicate over finite sequences.

To see, on an informal level, why bar recursion exists in continuous models, we first note that such models all satisfy the following sequential continuity principle:

$$\mathsf{Cont}\ :\ \forall \omega^{X^{\mathbb{N}} \to \mathbb{N}}, \alpha^{X^{\mathbb{N}}} \exists N \forall \beta([\alpha](N) =_{X^*} [\beta](N) \to \omega(\alpha) =_{\mathbb{N}} \omega(\beta)).$$

From this we can easily derive Spec: if N is a modulus of continuity on ω and α then $\hat{\omega}([\alpha](n)) < n$ holds for $n := \max\{N, \omega(\alpha) + 1\}$. Now, to show that $\mathsf{BR}^{\phi,b,\omega}_{X,R}(s)$ defines a total object for total arguments ϕ, b, ω and s, we argue by bar induction on the predicate

$$P(t) \equiv \mathsf{BR}^{\phi,b,\omega}_{X,R}(s * t) \text{ is total.}$$

[2] We use a variant of Kreisel's trick in the proof of Proposition 2.8 further below.

Given an infinite sequence $\alpha\colon X^{\mathbb{N}}$ it is clear by Spec that $\hat{\omega}(s*[\alpha](n)) < |s|+n$ for some n and therefore $\mathsf{BR}^{\phi,b,\omega}(s*[\alpha](n)) = b(s*[\alpha](n))$ is total. Clearly the bar induction step $\forall t^{X^*}(\forall x^X P(t*x) \to P(t))$ holds and thus we obtain $P(\langle\rangle)$ and therefore totality of $\mathsf{BR}^{\phi,b,\omega}_{X,R}(s)$. A broadly similar but somewhat more involved application of bar induction proves that BR exists in the majorizable functionals (see [9, 16]).

To summarise, the basic idea behind Spector's bar recursion is that any sequence of recursive calls made by BR eventually hits a *bar* s at which the condition $\omega(\hat{s}) < |s|$ holds and therefore $\mathsf{BR}(s)$ is assigned a value $b(s)$. These values propagate backwards along the tree of recursive calls ensuring that BR is defined everywhere.

2 A demand-driven variant of Spector's bar recursion

One can view Spector's bar recursion as just one instance of a more general form of backward recursion in which the main argument is some partial function with finite domain (for Spector a finite sequence s), and recursive calls are made by extending the domain of this argument (for Spector extending the sequence with one element $s*x$). From this perspective it seems that bar recursion is quite constrained in that the domain of its input is always an initial segment of the natural numbers. This has two obvious disadvantages. Firstly, the implicit dependence on the ordering of the natural numbers makes it unclear how to generalise BR to carry out recursion over partial functions on discrete structures which do not come equipped with a natural ordering. Secondly, adherence to sequentiality means that precise values of the control functional ω are never required: all that matters is whether or not $\omega(\hat{s}) < |s|$, or in other words, whether or not $\omega(\hat{s})$ is within the domain of already computed values. Thus when we consider that in terms of program extraction the parameter ω is typically some realizing term extracted from a lemma in a proof, Spector's bar recursor lacks sensitivity in that it ignores precise information from its proof-theoretic environment.

It is natural, then, to ask whether there is alternative to bar recursion which still terminates on inputs u with $\omega(\hat{u})$ in the domain of u, but which searches for these points in a more flexible way, and takes into account information provided by ω. This is the idea behind our variant of bar recursion, which we call *symmetric bar recursion*. The symmetric bar recursor $\mathsf{sBR}_{X,R}$ is given by the defining equation

$$\mathsf{sBR}^{\phi,b,\omega}_{X,R}(u^{X^\dagger}) =_R \begin{cases} b(u) & \text{if } \hat{\omega}(u) \in \mathrm{dom}(u) \\ \phi_u(\lambda x^X\,.\,\mathsf{sBR}^{\phi,b,\omega}(u \oplus (\hat{\omega}(u), x))) & \text{otherwise} \end{cases}$$

where now the parameters have type $\phi\colon X^\dagger \to (X \to R) \to R$, $b\colon X^\dagger \to R$ and $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$. Recall that the operation $\oplus$ indicates the extension of the partial function u with one more piece of information, analogous to the extension of finite partial functions in the defining equation of BR. The crucial difference is that this extension can potentially take place at any point $n \in \mathbb{N}\backslash\mathrm{dom}(u)$, and so we are no longer restricted to making recursive calls in a sequential fashion. However, this additional freedom requires us to carefully justify the totality of sBR, as its recursive calls are not easily seen to be well-founded. In Definition 2.6 below we give a corresponding symmetric bar induction principle which can be used to reason about sBR. First we need the following important definition.

Definition 2.1 (Finite ω-threads of $u\colon X^\dagger$ or $\alpha\colon X^{\mathbb{N}}$). Given $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$ and $u\colon X^\dagger$, the *ω-thread of u of length i* is the finite partial function $\{u\}_\omega(i)\colon X^\dagger$ inductively defined as

$$\{u\}_\omega(0) := \emptyset$$
$$\{u\}_\omega(i+1) := \begin{cases} \{u\}_\omega(i) \oplus (n_{\omega,i}, u(n_{\omega,i})) & \text{if } n_{\omega,i} \in \mathrm{dom}(u) \\ \{u\}_\omega(i) & \text{otherwise} \end{cases}$$

where $n_{\omega,i} := \hat{\omega}(\{u\}_\omega(i))$. Note that when either $n_{\omega,i} \in \mathrm{dom}(\{u\}_\omega(i))$ or $n_{\omega,i} \notin \mathrm{dom}(u)$, we would have $\{u\}_\omega(j) = \{u\}_\omega(i)$ for all $j \geq i$. Entirely analogously, define the *ω-thread of α of length i*, also denoted $\{\alpha\}_\omega(i)\colon X^\dagger$, as

$$\{\alpha\}_\omega(0) := \emptyset$$
$$\{\alpha\}_\omega(i+1) := \{\alpha\}_\omega(i) \oplus (n_{\omega,i}, \alpha(n_{\omega,i}))$$

where $n_{\omega,i} := \hat{\omega}(\{\alpha\}_\omega(i))$.

Remark 2.2. In what follows we will frequently just write $\{u\}(i)$ when ω is clear from the context.

Definition 2.3 (ω-threads). Let $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$. We say that a finite partial function $u\colon X^\dagger$ is a *ω-thread* if $u = \{u\}_\omega(|\mathrm{dom}(u)|)$. This can be expressed formally by the decidable predicate

$$S_\omega(u) :\equiv \forall n(n \in \mathrm{dom}(u) \to n \in \mathrm{dom}(\{u\}_\omega(|\mathrm{dom}(u)|))).$$

The following lemma shows that for all i, $n \in \mathrm{dom}(\{u\}_\omega(i))$ implies that $\{u\}_\omega(i)(n) = u(n)$, and hence $S_\omega(u) \Leftrightarrow u = \{u\}_\omega(|\mathrm{dom}(u)|)$. The intuition here is that ω works as a control function that dictates the location of the next bar recursive call. Thus ω-threads are just partial functions that have been constructed using ω as a control functional.

Lemma 2.4. *Suppose that $n_{\omega,i}$ is defined as in Definition 2.1. A finite partial function u satisfies $S_\omega(u)$ (i.e. is a ω-thread) iff for all $i \leq |\mathrm{dom}(u)|$,*

$$\{u\}_\omega(i) = (n_{\omega,0}, x_0) \oplus (n_{\omega,1}, x_1) \oplus \ldots \oplus (n_{\omega,i-1}, x_{i-1})$$

for $x_j = u(n_{\omega,j})$, where the $n_{\omega,j}$ are all distinct and all lie in $\mathrm{dom}(u)$*. In particular*

$$S_\omega(u) \Rightarrow u = \{u\}_\omega(l) = (n_{\omega,0}, x_0) \oplus \ldots \oplus (n_{\omega,l-1}, x_{l-1})$$

where $l = |\mathrm{dom}(u)|$.

Proof. For one direction, assume $S_\omega(u)$ and set $l := |\mathrm{dom}(u)|$. We use induction on $i \leq l$. If $i = 0$ then $\{u\}_\omega(0) = \emptyset$ by definition. Now for $i < l$ assume that $\{u\}_\omega(i) = (n_{\omega,0}, x_0) \oplus \ldots \oplus (n_{\omega,i-1}, x_{i-1})$ for distinct $n_{\omega,j}$. If either $n_{\omega,i} \notin \mathrm{dom}(u)$ or $n_{\omega,i} \in \{n_{\omega,0}, \ldots, n_{\omega,i-1}\}$ then, by the definition of the ω-thread of u, we would have that $\{u\}_\omega(i) = \{u\}_\omega(i+1) = \ldots = \{u\}_\omega(l)$, and hence $i = |\mathrm{dom}(\{u\}_\omega(i))| = |\mathrm{dom}(\{u\}_\omega(l))| < l$, contradicting $S_\omega(u)$. Therefore, we must have $n_{\omega,i} \in \mathrm{dom}(u) \backslash \{n_{\omega,0}, \ldots, n_{\omega,i-1}\}$ and

$$\{u\}_\omega(i+1) = (n_{\omega,0}, x_0) \oplus \ldots \oplus (n_{\omega,i-1}, x_{i-1}) \oplus (n_{\omega,i}, x_i)$$

for $x_i = u(n_{\omega,i})$, where the $n_{\omega,j}$ are all distinct and belong to $\mathrm{dom}(u)$. The other direction is straightforward: If $\{u\}_\omega(l) = (n_{\omega,0}, x_0) \oplus \ldots \oplus (n_{\omega,l-1}, x_{l-1})$ for distinct $n_{\omega,j}$ then $\mathrm{dom}(\{u\}_\omega(l)) = l = \mathrm{dom}(u)$. But since $\{u\}_\omega(l)$ is only defined at points where u is defined, and at those points they hold the same value, it means that $u = \{u\}_\omega(l)$, i.e. $S_\omega(u)$. □

Example 2.5. Define $u := (1,1) \oplus (2,2) \oplus (3,3)$ so that u is the partial identity function defined at $1, 2, 3$.

1. Let $\omega(\alpha) := \max\{\alpha(0), \alpha(1), \alpha(2)\} + 1$. We have that $n_{\omega,0} = \hat{\omega}(\emptyset) = \max\{0,0,0\} + 1 = 1$ and thus $\{u\}_\omega(1) = (1,1)$, and similarly $n_{\omega,1} = \max\{0,1,0\} + 1 = 2$, $\{u\}_\omega(2) = (1,1) \oplus (2,2)$, and $n_{\omega,2} = \max\{0,1,2\} + 1 = 3$, $\{u\}_\omega(3) = (1,1) \oplus (2,2) \oplus (3,3) = u$. Hence, u is a ω-thread.
2. On the other hand, for $\psi(\alpha) := \max\{\alpha(0), \alpha(1), \alpha(2)\}$ we have $n_{\psi,0} = 0 \notin \mathrm{dom}(u)$ and thus $\{u\}_\psi(i) = \emptyset$ for all i, so u is not a ψ-thread.

Definition 2.6. Let us write $\forall u \in S_\omega\, A(u)$ as an abbreviation for $\forall u(S_\omega(u) \to A(u))$. The principle of *symmetric bar induction* sBI is given by

$$\mathsf{sBI}:\ \forall \omega^{X^{\mathbb{N}} \to \mathbb{N}} \Big(\forall \alpha^{X^{\mathbb{N}}} \exists n P(\{\alpha\}_\omega(n)) \wedge \forall u \in S_\omega([\omega(\hat{u}) \notin \mathrm{dom}(u) \to \forall x^X P(u \oplus (\omega(\hat{u}), x))] \to P(u)) \to P(\emptyset) \Big)$$

where P is an arbitrary predicate on $X^\dagger$.

Theorem 2.7. $\mathsf{E\text{-}PA}^\omega + \mathsf{DC} \vdash \mathsf{sBI}$.

Proof. Fix some ω and suppose for a contradiction that the premises of sBI are true but $\neg P(\emptyset)$. The second premise of sBI is classically equivalent to

$$\forall u \in S_\omega(\neg P(u) \to [\omega(\hat{u}) \notin \mathrm{dom}(u) \wedge \exists x \neg P(u \oplus (\omega(\hat{u}), x))]).$$

Hence, by dependent choice, there exists a sequence $u_0, u_1, \ldots$ of finite partial functions, together with a sequence $x_0, x_1, \ldots$ of elements of X, satisfying

$$u_0 = \emptyset \quad \text{and} \quad u_{i+1} = u_i \oplus (n_i, x_i)$$

with $n_i = \hat{\omega}(u_i) \notin \mathrm{dom}(u_i)$. Clearly each u_i is a ω-thread, i.e. $S_\omega(u_i)$. Moreover, by construction we have that $\neg P(u_i)$ holds for all i. Now, by classical countable choice there exists a function $\alpha\colon X^{\mathbb{N}}$ satisfying

$$\alpha(n) := \begin{cases} u_i(n) & \text{where } i \text{ is the least such that } n \in \mathrm{dom}(u_i), \text{ if it exists} \\ \mathbf{0}_X & \text{otherwise.} \end{cases}$$

Let us first show by induction on i that $\{\alpha\}_\omega(i) = u_i$, for all i. First, $\{\alpha\}_\omega(0) = \emptyset$ by definition. Assuming that $\{\alpha\}_\omega(i) = u_i$ we have $\hat{\omega}(\{\alpha\}_\omega(i)) = \hat{\omega}(u_i) = n_i$, and therefore $\{\alpha\}_\omega(i+1) = \{\alpha\}_\omega(i) \oplus (n_i, \alpha(n_i)) = u_i \oplus (n_i, \alpha(n_i))$. Now by construction $n_i \notin \mathrm{dom}(u_i)$ and $n_i \in \mathrm{dom}(u_{i+1})$. Thus $\alpha(n_i) = u_{i+1}(n_i) = x_i$ and therefore $\{\alpha\}_\omega(i+1) = u_{i+1}$. That concludes the proof that $\{\alpha\}_\omega(i) = u_i$.
By the first premise of sBI there exists some n such that $P(\{\alpha\}_\omega(n))$, which implies $P(u_n)$, contradicting the assumption that $\neg P(u_i)$ holds for all i. □

Intuitively, symmetric bar recursion is well-founded only if every sequence of recursive calls eventually arrives at some u satisfying $\hat{\omega}(u) \in \mathrm{dom}(u)$. Put formally, this statement forms a symmetric analogue of Spec, namely

$$\mathsf{sSpec}_X \ : \ \forall \omega^{X^{\mathbb{N}} \to \mathbb{N}} \forall \alpha^{X^{\mathbb{N}}} \exists n (\hat{\omega}(\{\alpha\}_\omega(n)) \in \mathrm{dom}(\{\alpha\}_\omega(n))).$$

We can in fact express the relationship between the recursor sBR and the logical principle sSpec in more concrete terms by adapting Proposition 1.1 from [14]: namely we can prove that sSpec must be valid in any model of $\mathsf{E\text{-}HA}^\omega + \mathsf{sBR}$, which we do in Proposition 2.8 below. Conversely, in Theorem 2.11 we use the fact that sSpec is satisfied in the model $\mathcal{C}^\omega$ of total continuous functionals to justify the existence of sBR in this model.

Proposition 2.8. *Define the term $\theta_{\omega,\alpha}$ in $\mathsf{E\text{-}HA}^\omega + (\mathsf{sBR})$ with free variables $\alpha\colon X^{\mathbb{N}}$ and $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$ by*

$$\theta_{\omega,\alpha}(u^{X^\dagger}) = \begin{cases} 0 & \text{if } \omega(\hat{u}) \in \mathrm{dom}(u) \\ 1 + \theta_{\omega,\alpha}(u \oplus (\omega(\hat{u}), \alpha(\omega(\hat{u})))) & \text{otherwise.} \end{cases}$$

Then, provably in $\mathsf{E\text{-}HA}^\omega + (\mathsf{sBR})$, we have $\hat{\omega}(\{\alpha\}_\omega(n)) \in \mathrm{dom}(\{\alpha\}_\omega(n))$ for some $n \leq \theta_{\omega,\alpha}(\emptyset)$.

Proof. Fix $\alpha\colon X^{\mathbb{N}}$ and $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$. Let $\beta i := \theta_{\omega,\alpha}(\{\alpha\}_\omega(i))$. By definition of $\theta_{\omega,\alpha}$ we have

$$\beta i = \begin{cases} 0 & \text{if } \hat{\omega}(\{\alpha\}_\omega(i)) \in \mathrm{dom}(\{\alpha\}_\omega(i)) \\ 1 + \beta(i+1) & \text{otherwise.} \end{cases}$$

First note that, by the definition of β, we have

$$(*) \quad \beta i \neq 0 \quad \text{iff} \quad \hat{\omega}(\{\alpha\}_\omega(i)) \notin \mathrm{dom}(\{\alpha\}_\omega(i)).$$

By induction on i, using $(*)$, it is easy to show

$$\forall i (\forall j \leq i \, (\beta j \neq 0) \to \forall j \leq i \, (\beta j = 1 + \beta(j+1))).$$

By another induction on i, using the above fact, we obtain

$$\forall i (\forall j < i \, (\beta j \neq 0) \to \beta 0 = i + \beta i).$$

Therefore, setting $i = \beta 0$ we have

$$\forall j < \beta 0 (\beta j \neq 0) \to \beta 0 = \beta 0 + \beta(\beta 0).$$

Therefore either $\beta(\beta 0) = 0$, or $\exists j < \beta 0 (\beta j = 0)$, i.e. $\exists j \leq \beta 0 (\beta j = 0)$. Using $(*)$ we have

$$\exists j \leq \beta 0 (\hat{\omega}(\{\alpha\}_\omega(j)) \in \mathrm{dom}(\{\alpha\}_\omega(j))).$$

That concludes the proof since $\beta 0 = \theta_{\omega,\alpha}(\{\alpha\}_\omega(0)) = \theta_{\omega,\alpha}(\emptyset)$. □

2.1 Relating Spec and sSpec

We now make our first link between symmetric bar recursion and Spector's bar recursion via their corresponding axioms sSpec and Spec.

Theorem 2.9. $\mathsf{E\text{-}HA}^\omega + \mathsf{sSpec}_{X\times\mathbb{B}} \vdash \mathsf{Spec}_X$

Proof. Given $\alpha\colon X^{\mathbb{N}}$ and $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$ we need to produce a point n such that $\hat{\omega}([\alpha](n)) < n$. Recall that $\mathbb{B} = \{0,1\}$ is the type of Booleans. Define $\tilde{\alpha}\colon (X\times\mathbb{B})^{\mathbb{N}}$ and $\theta\colon (X\times\mathbb{B})^{\mathbb{N}} \to \mathbb{N}$ in terms of α and ω as

$$\tilde{\alpha}(n) := \langle \alpha(n), 1\rangle,$$

$$\theta(\beta) := \mu i \leq \omega(\lambda k.\pi_0(\beta k))(\pi_1(\beta i) =_{\mathbb{B}} 0)$$

where $\pi_0\colon X\times\mathbb{B} \to X$ and $\pi_1\colon X\times\mathbb{B} \to \mathbb{B}$ are the two projections, and μ is the bounded search operator. Intuitively, we are using the booleans to indicate whether a position is 'defined' (i.e. equal to 1) or not. Hence, the functional θ returns the first undefined position of β which is bounded by $\omega(\lambda k.\pi_0(\beta k))$, or just 0 if no such position is found. By sSpec there exists some N such that

(i) $\hat{\theta}(\{\tilde{\alpha}\}_\theta(N)) \in \mathrm{dom}(\{\tilde{\alpha}\}_\theta(N))$.

Without loss of generality let N be the least such value. We will show that $\hat{\omega}([\alpha](N)) < N$. To do this, we first claim that

(ii) $\forall m \leq N(\mathrm{dom}(\{\tilde{\alpha}\}_\theta(m)) = \{0,\ldots,m-1\})$ and $\forall m < N(\hat{\omega}([\alpha](m)) \geq m)$.

The proof of (ii) is by induction on m. If $m = 0$ the claim is trivial. Now assume that (ii) holds for m. For the first part, suppose that $m < N$. Then by the induction hypothesis we have

$$\begin{aligned}\hat{\theta}(\{\tilde{\alpha}\}_\theta(m)) &= \theta(\langle\alpha(0),1\rangle,\ldots,\langle\alpha(m-1),1\rangle,\langle\mathbf{0},0\rangle,\langle\mathbf{0},0\rangle,\ldots)\\ &= \mu i \leq \hat{\omega}([\alpha](m))(i \geq m)\\ &= m\end{aligned} \quad ,$$

using that $m < N$ and thus $\hat{\omega}([\alpha](m)) \geq m$ by the second induction hypothesis. Therefore by definition we have $\{\tilde{\alpha}\}_\theta(m+1) = \{\tilde{\alpha}\}_\theta(m) \oplus (m, \tilde{\alpha}(m))$, and thus $\mathrm{dom}(\{\tilde{\alpha}\}_\theta(m+1)) = \{0,\ldots,m\}$.
For the second part, suppose for a contradiction that $m < N$ but $\hat{\omega}([\alpha](m)) < m$. Then by the first part we have

$$\hat{\theta}(\{\tilde{\alpha}\}_\theta(m)) \leq \hat{\omega}([\alpha](m)) < m$$

which would imply that $\hat{\theta}(\{\tilde{\alpha}\}_\theta(m)) \in \{0,\ldots,m-1\}$, contradicting the assumed minimality of N. Therefore we have established (ii), and setting $m = N$ we have $\mathrm{dom}(\{\tilde{\alpha}\}_\theta(N)) = \{0,\ldots,N-1\}$. By (i), this of course implies that $\hat{\theta}(\{\tilde{\alpha}\}_\theta(N)) < N$, and unwinding the definition θ we obtain

$$N > \hat{\theta}(\{\tilde{\alpha}\}_\theta(N)) = \mu i \leq \hat{\omega}([\alpha](N))(i \geq N).$$

If $\hat{\omega}([\alpha](N)) \geq N$ then the unbounded search would select N. So we must have that $\hat{\omega}([\alpha](N)) < N$. □

Theorem 2.10. $\mathsf{E\text{-}PA}^\omega + \mathsf{AC}_{\mathbb{N},X} + \mathsf{Spec}_{X^\dagger} \vdash \mathsf{sSpec}_X$.

Proof. Let $\alpha\colon X^{\mathbb{N}}$ and $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$ be given. We must find n such that $\hat{\omega}(\{\alpha\}_\omega(n)) \in \mathrm{dom}(\{\alpha\}_\omega(n))$. Using $\mathsf{AC}_{\mathbb{N},X}$ and classical logic we can define the sequence of indices $(i_n)_{n\in\mathbb{N}}$ as

$$i_n := \begin{cases} i & \text{where } i \text{ is the least such that } n \in \mathrm{dom}(\{\alpha\}_\omega(i)), \text{ if such } i \text{ exists}\\ 0 & \text{if no such } i \text{ exists.}\end{cases}$$

Using $(i_n)_{n\in\mathbb{N}}$ we can then define the sequence $\tilde{\alpha}\colon (X^\dagger)^{\mathbb{N}}$ as

$$\tilde{\alpha}(n) := \{\alpha\}_\omega(i_n.)$$

Note that $\tilde{\alpha}$ represents a characteristic function in the following sense:

$$(*) \quad n \in \mathrm{dom}(\tilde{\alpha}(n)) \Leftrightarrow \exists i(n \in \mathrm{dom}(\{\alpha\}_\omega(i)))$$

Next, we primitive recursively define the 'diagonalisation' function $d\colon (X^\dagger)^{\mathbb{N}} \to (X+\mathbf{1})^{\mathbb{N}}$ by

$$d(\beta)(j) :=_{X+\mathbf{1}} \begin{cases} \beta(k)(j) & \text{for least } k \leq j \text{ such that } j \in \mathrm{dom}(\beta(k)) \\ \text{undefined} & \text{if no such } k \leq j \text{ exists.} \end{cases}$$

Finally, we define the functional $\theta\colon (X^\dagger)^{\mathbb{N}} \to \mathbb{N}$ by

$$\theta(\beta) := \hat{\omega}(d(\beta)).$$

Now, applying $\mathsf{Spec}_{X^\dagger}$ to θ and $\tilde{\alpha}$ obtain a number N such that

(i) $\hat{\theta}([\tilde{\alpha}](N)) < N$.

Let i_m be the maximum number in the set $\{i_0, \ldots, i_{N-1}\}$. We claim that i_m is our desired witness, i.e. $\hat{\omega}(\{\alpha\}_\omega(i_m)) \in \mathrm{dom}(\{\alpha\}_\omega(i_m))$. First, let $\mathrm{emb}(\cdot)\colon X^\dagger \to (X+\mathbf{1})^{\mathbb{N}}$ denote the embedding of finite partial functions into the type of arbitrary partial sequences. We prove that

(ii) $\hat{d}([\tilde{\alpha}](N)) = \mathrm{emb}(\tilde{\alpha}(m))$.

We consider two cases:
If $j \notin \mathrm{dom}(\hat{d}([\tilde{\alpha}](N)))$ then by definition of d we have that $\neg\exists k \leq j(j \in \mathrm{dom}(\widehat{[\tilde{\alpha}](N)}(k)))$. We consider a further two cases. If $j < N$ then $j \notin \mathrm{dom}(\tilde{\alpha}(k))$ for all $k \leq j$, which in particular implies $j \notin \mathrm{dom}(\tilde{\alpha}(j))$ and thus by $(*)$ we get $j \notin \mathrm{dom}(\{\alpha\}_\omega(i_m)) = \tilde{\alpha}(m)$. If $j \geq N$ then $j \notin \mathrm{dom}(\tilde{\alpha}(k))$ for all $k < N$, which in particular implies $j \notin \mathrm{dom}(\tilde{\alpha}(m))$.
If $j \in \mathrm{dom}(d(\widehat{[\tilde{\alpha}](N)}))$ we know that $j \in \mathrm{dom}(\tilde{\alpha}(k))$ for some $k \leq j$ (with $k < N$) and $\hat{d}([\tilde{\alpha}](N))(j) = \tilde{\alpha}(k)(j) = \alpha(j)$, and since $i_k \leq i_m$ we have $\tilde{\alpha}(k) \sqsubseteq \tilde{\alpha}(m)$ and thus $\tilde{\alpha}(m)(j) = \alpha(j)$, which concludes the proof of (ii).
Therefore, using our usual abbreviation $n_{\omega,k} = \hat{\omega}(\{\alpha\}_\omega(k))$, we obtain that $n_{\omega,i_m} < N$ as follows:

$$n_{\omega,i_m} = \hat{\omega}(\{\alpha\}_\omega(i_m)) = \hat{\omega}(\mathrm{emb}(\{\alpha\}_\omega(i_m))) = \hat{\omega}(\mathrm{emb}(\tilde{\alpha}(m))) \overset{(ii)}{=} \hat{\omega}(d(\widehat{[\tilde{\alpha}](N)})) = \theta(\widehat{[\tilde{\alpha}](N)}) \overset{(i)}{<} N.$$

Now, to prove the main result, suppose for a contradiction that $n_{\omega,i_m} \notin \mathrm{dom}(\{\alpha\}_\omega(i_m))$. Then by the definition of $\{\alpha\}_\omega(i)$ we have $n_{\omega,i_m} \in \mathrm{dom}(\{\alpha\}_\omega(i_m + 1))$, and moreover $i_m + 1$ is the least such index and we have $i_{n_{\omega,i_m}} = i_m + 1$. But by maximality of i_m for the set $\{i_0, \ldots, i_{N-1}\}$ and the fact that $n_{\omega,i_m} < N$ we have $i_m \geq i_{n_{\omega,i_m}} = i_m + 1$, a contradiction. □

In section 5 we expand the ideas in the proofs of Theorems 2.9 and 2.10 to show that the recursors BR and sBR themselves are primitive recursively equivalent. This in particular implies the following:

Theorem 2.11. *The Kleene/Kreisel continuous functionals $\mathcal{C}^\omega$ are a model of* sBR.

Proof. In Theorem 5.2 we show that there is a term in $\mathcal{T} :\equiv \mathsf{E\text{-}HA}^\omega + \mathsf{sBI} + \mathsf{BR}$ which satisfies the defining equation of sBR provably in $\mathcal{T}$. Making use of this inter-definability result, the theorem follows directly: By Theorem 2.7, we can reduce sBI to DC over $\mathsf{E\text{-}PA}^\omega$, and thus $\mathcal{T}$ can be reduced to $\mathsf{E\text{-}PA}^\omega + \mathsf{DC} + \mathsf{BR}$. But it is well-known (see e.g. [25]) that $\mathcal{C}^\omega$ is a model of $\mathsf{E\text{-}PA}^\omega + \mathsf{DC} + \mathsf{BR}$, and therefore we can conclude that sBR exists in $\mathcal{C}^\omega$.

However, since the proof of Theorem 5.2 is rather intricate, and we wish to make the present section self-contained, we present here a more direct domain-theoretic argument for the existence of sBR in $\mathcal{C}^\omega$. It is a standard result [10] that the total continuous functionals $\mathcal{C}^\omega$ are just the extensional collapse of the partial continuous functionals $\hat{\mathcal{C}}^\omega$. Now sBR can be easily defined in $\hat{\mathcal{C}}^\omega$ as the least fixpoint Φ of its recursive defining equation, since $\hat{\mathcal{C}}^\omega$ has the property that all recursive functionals $Z \to Z$ have a fixed point $p\colon Z$. Now it is well-known that all *total* continuous functionals ω, α satisfy Spector's property Spec, and thus adapting Theorem 2.10 to the total continuous functionals, we have that sSpec also holds for all total ω, α. We can then prove that the fixpoint Φ is a total function using sBI, which is also valid in continuous models since as shown in Theorem 2.7 it follows directly from DC. Suppose that the parameters ϕ, ω and b and an argument v of Φ are total, and let

$$P(u^{X^\dagger}) :\equiv \Phi(v \oplus u) \text{ is total.}$$

Then by sSpec on the total functional $\psi(\alpha) := \omega(v \oplus \alpha)$ we have that $\forall\alpha\exists n(\hat{\psi}(\{\alpha\}_\psi(n)) \in \mathrm{dom}(\{\alpha\}_\psi(n)))$. By the definition of ψ this implies

$$\forall\alpha\exists n(\hat{\omega}(v \oplus \{\alpha\}_\omega(n)) \in \mathrm{dom}(v \oplus \{\alpha\}_\omega(n)))$$

and hence $\forall\alpha\exists n P(\{\alpha\}_\omega(n))$, the first hypothesis of sBI. For the second hypothesis of sBI, we assume that for $u \in S_\psi$ we have $\psi(\hat{u}) \notin \mathrm{dom}(u) \to \forall x^X P(u \oplus (\psi(\hat{u}), x))$, and aim to prove that $\Phi(v \oplus u)$ is total. If $\Phi(v \oplus u)$ was not total then obviously we would have that $\hat{\psi}(u) \notin \mathrm{dom}(v \oplus u)$, which implies $\hat{\psi}(u) \notin \mathrm{dom}(u)$. But in this case $\forall x P(u \oplus (\hat{\psi}(u), x)))$ implies that $\lambda x.\Phi(v \oplus u \oplus (\hat{\omega}(v \oplus u), x))$ is total, and thus so is $\Phi(v \oplus u)$. Therefore since both premises of sBI hold, we can conclude $P(\emptyset)$ i.e. $\Phi(v)$ is total. □

3 The Dialectica interpretation of countable choice

In the following sections we assume that the reader is broadly familiar with Gödel's Dialectica interpretation and its role in the extraction of computational content from proofs – details of which can be found in e.g. [3, 16] – although we make an effort to keep the main flow of ideas as self-contained as possible.

Recall that the Dialectica interpretation translates each formula A in the language of some theory $\mathcal{T}$ to a quantifier-free formula $|A|^x_y$ in some 'verifying' functional theory $\mathcal{S}$, where x and y are (possibly empty) tuples of variables of some finite type. The idea is that A is (classically) equivalent to $\exists x \forall y |A|^x_y$, and that the interpretation $\mathcal{T} \to \mathcal{S}$ is sound if whenever $\mathcal{T} \vdash A$ we can extract a realizer t for $\exists x$ so that $\mathcal{S} \vdash |A|^t_y$. When $\mathcal{T}$ is a classical theory, one typically precomposes the Dialectica interpretation with a negative translation in order to obtain soundness, a combination normally referred to as the *ND interpretation*.

The Dialectica interpretation was conceived by Gödel in the 1930s, and published much later in a seminal paper of 1958 [13] in which it was shown that Peano arithmetic can be ND interpreted into the system T of primitive recursive functionals in all finite types. In fact it is not too difficult to lift Gödel's soundness proof to the higher-type theory $\mathsf{WE\text{-}PA}^\omega + \mathsf{QF\text{-}AC}$ of *weakly-extensional* Peano arithmetic with the quantifier-free axiom of choice (see [16] for details). On the other hand, for the addition of computationally non-trivial choice principles, such as the axiom of countable choice $\mathsf{AC}_\mathbb{N}$, for arbitrary formulas, the primitive recursive functionals no longer suffice for soundness of the interpretation. In fact, over $\mathsf{WE\text{-}PA}^\omega$ countable choice is strong enough to derive the full comprehension schema

$$\mathsf{CA} \ : \ \exists f^{\mathbb{N}\to X} \forall n (f(n) = 0 \leftrightarrow A(n))$$

and so the theory $\mathsf{WE\text{-}PA}^\omega + \mathsf{QF\text{-}AC} + \mathsf{AC}_\mathbb{N}$ is already capable of formalising a large portion of mathematical analysis, and is thus considerably stronger than Peano arithmetic. Nevertheless, just a few years after Gödel's paper, Spector [24] proved that one could indeed extend the Dialectica interpretation to full classical analysis provided we add bar recursion to system T.

3.1 The countable choice problem

Spector's main idea can be appreciated from a completely abstract perspective, independently of the full details of the Dialectica interpretation. Spector observed that in order to extend the ND interpretation to $\mathsf{WE\text{-}PA}^\omega + \mathsf{QF\text{-}AC} + \mathsf{AC}_\mathbb{N}$, it suffices to find some way of realizing the Dialectica interpretation of the double negation shift:

$$\mathsf{DNS} \ : \ \forall n \neg\neg B(n) \to \neg\neg \forall n B(n).$$

Now, suppose that the Dialectica interpretation of $B(n)$ is $|B(n)|^x_y$ where $x\colon X$ and $y\colon Y$ are tuples of variables of the appropriate type. Then the Dialectica interpretation of DNS is given by

$$|\mathsf{DNS}|^{f,p,n}_{\varepsilon,q,\omega} = |B(n)|^{\varepsilon_n p}_{p(\varepsilon_n p)} \to |B(\omega f)|^{f(\omega f)}_{qf}.$$

In other words, to solve the Dialectica interpretation of DNS, for each given formula B one must produce realizers $f\colon X^\mathbb{N}$, $p\colon X \to Y$ and $n\colon \mathbb{N}$ in terms of the parameters $\varepsilon\colon \mathbb{N} \to (X \to Y) \to X$, $q\colon X^\mathbb{N} \to Y$ and $\omega\colon X^\mathbb{N} \to \mathbb{N}$ satisfying $|\mathsf{DNS}|^{f,p,n}_{\varepsilon,q,\omega}$. Spector approached this by tackling a stronger problem, namely to solve the underlying system of equations

$$\begin{aligned} \omega f &= n \\ f n &= \varepsilon_n p \\ q f &= p(\varepsilon_n p) \end{aligned} \tag{1}$$

in f, p and n. Indeed, given any solution f, p and n for the system of equations above, we have that

$$|B(n)|^{\varepsilon_n p}_{p(\varepsilon_n p)} \to |B(\omega f)|^{f(\omega f)}_{qf}$$

is logically true. One might wonder what the precise relation is between these constructed functionals f, p, n and the actual axiom of countable choice

$$\mathsf{AC}_{\mathbb{N},X} : \forall n^{\mathbb{N}} \exists x^X A_n(x) \to \exists f^{\mathbb{N}\to X} \forall n A_n(f(n)).$$

Although the precise connection goes through the ND-interpretation, a helpful intuition is that although we cannot effectively construct the functional f witnessing $\mathsf{AC}_{\mathbb{N},X}$, it is possible to construct an approximation to f which is correct at a given point n. But that point n is not fixed, but is only known once we have constructed f itself, via the control functional ω. It is the breaking of this circularity that is the job of bar recursion[3].

We call the equations (1) *Spector's equations*, and the issue of solving them the *countable choice problem*. It is clear that a solution Spector's equations is also a realizer for DNS, even independent of the formula B, and thus to extend the ND interpretation to classical analysis it suffices to find a general solution to the countable choice problem. In the following two sub-section we present both Spector's solution to the countable choice problem and a new solution based on the symmetric bar recursor.

3.2 Spector's bar recursive solution

Spector's classic solution to the countable choice problem was to use the general bar recursion BR to construct a term $\Phi^{\varepsilon,q,\omega}_X$ in the parameters ε, q and ω of the problem, which satisfies the defining equation

$$\Phi^{\varepsilon,q,\omega}_X(s) =_{X^*} s \oplus \begin{cases} \langle\rangle & \text{if } \omega(\hat{s}) < |s| \\ \Phi^{\varepsilon,q,\omega}_X(s * a_s) & \text{otherwise.} \end{cases}$$

where $a_s := \varepsilon_{|s|}(\lambda x.\hat{q}(\Phi^{\varepsilon,q,\omega}_X(s * x)))$. Actually Spector's defines a slightly different (but equivalent) variant of $\Phi^{\varepsilon,q,\omega}_X(s)$, but precise details are not particularly relevant here (see e.g. [19]).

Theorem 3.1 ([24])**.** *Define*

$$\begin{aligned} t &:=_{X^*} \Phi^{\varepsilon,q,\omega}_X(\langle\rangle) \\ p_i &:=_{X\to Y} \lambda x.\hat{q}(\Phi^{\varepsilon,q,\omega}_X([t](i) * x)) \end{aligned}$$

where $i < |t|$ in the second equation. Then for all $0 \le i < |t|$ we have

$$\begin{aligned} t_i &= \varepsilon_i p_i \\ \hat{q}t &= p_i(\varepsilon_i p_i). \end{aligned} \tag{2}$$

Proof. This is a standard induction argument see e.g. [16, 18]. □

Corollary 3.2. *Define t and p_i in the parameters ε, q and ω as in Theorem 3.1. Then $f := \hat{t}$, $n := \omega f$, and $p := p_n$ solve Spector's equations (1).*

Proof. The proof essentially reduces to verifying that $\omega(\hat{t}) < |t|$. The result then follows directly from Theorem 3.1 and the equations (2). A sketch of the argument is as follows. Note that $\Phi(s)$ is an extension of s and hence $|\Phi(s)| \ge |s|$. One first shows that if $t = \Phi(\langle\rangle)$ then $\Phi(\langle\rangle) = \Phi(t)$. Hence, if $\omega(\hat{t}) \ge |t|$ we would have that $\Phi(t) = t \oplus \Phi(t * a_t)$ and thus $|t| = |\Phi(\langle\rangle)| = |\Phi(t)| = |t \oplus \Phi(t * a_t)| = |t \oplus (t \oplus a_t * \ldots)| \ge |t * a_t| = |t| + 1$, a contradiction. □

3.3 A symmetric solution

We now present our alternative solution to the countable choice problem which is based on our symmetric bar recursor sBR instead of the usual Spector recursor BR. Our first step is to define a symmetric version

[3]For more intuition on Spector's bar recursion, the interpretation of countable choice and Spector's equations see [18].

of the special recursor $\Phi_X^{\varepsilon,q,\omega}(s)$, which takes parameters ε, q and ω of the same type as those in for Φ, but whose input and output are now *finite partial functions*:

$$\Psi_X^{\varepsilon,q,\omega}(u^{X^\dagger}) =_{X^\dagger} u \oplus \begin{cases} \emptyset & \text{if } n_u \in \mathrm{dom}(u) \\ \Psi_X^{\varepsilon,q,\omega}(u \oplus (n_u, a_u)) & \text{otherwise} \end{cases}$$

where $n_u = \omega(\hat{u})$ and $a_u = \varepsilon_{n_u}(\lambda x.\hat{q}(\Psi_X^{\varepsilon,q,\omega}(u \oplus (n_u, x))))$. We note without proof that Ψ is indeed definable from sBR:

Proposition 3.3. *The functional* $\mathsf{sBR}_{X,X^\dagger}^{\phi^{\varepsilon,q,\omega},\lambda\alpha.\alpha,\omega}(u)$*, where*

$$\phi_u^{\varepsilon,q,\omega}(p^{X\to X^\dagger}) :=_{X^\dagger} u \oplus p(\varepsilon_{\omega(\hat{u})}(\lambda x.\hat{q}(p(x)))),$$

satisfies the defining equation of $\Psi_X^{\varepsilon,q,\omega}(u)$*, provably in* $\mathsf{E\text{-}HA}^\omega$*.*

Our construction and verification of a solution to Spector's equations now broadly follows, but is somewhat more intricate than, the usual approach for Spector's standard bar recursion. Recall the notion of a ω-thread from Definition 2.3.

Lemma 3.4. *Assume* u *is a* ω*-thread*[4]*. Let* $v := \Psi_X^{\varepsilon,q,\omega}(u)$*. Then*

$$v = \Psi_X^{\varepsilon,q,\omega}(\{v\}_\omega(i)) \tag{3}$$

for all $|\mathrm{dom}(u)| \le i \le |\mathrm{dom}(v)|$*.*

Proof. By induction on i.
For $i = |\mathrm{dom}(u)|$ we claim that $\{v\}_\omega(i) = u$. By a separate easy induction on $j \le |\mathrm{dom}(u)|$, one can show that whenever $u \sqsubseteq v$ and $S_\omega(u)$ then $\{u\}_\omega(j) = \{v\}_\omega(j)$ for all $0 \le j \le |\mathrm{dom}(u)|$. Now in this case $u \sqsubseteq v$ by definition, and thus setting $j = |\mathrm{dom}(u)|$ we have $\{v\}_\omega(|\mathrm{dom}(u)|) = u$.
Now, for the main induction step, assume that (3) is true for some $|\mathrm{dom}(u)| \le i < |\mathrm{dom}(v)|$. Because i is strictly smaller than $|\mathrm{dom}(v)|$ it must be the case that $n_{\{v\}_\omega(i)} := \hat{\omega}(\{v\}_\omega(i)) \notin \mathrm{dom}(\{v\}_\omega(i))$. Therefore, by the defining equation of Ψ we obtain

$$v \overset{\text{I.H.}}{=} \Psi(\{v\}_\omega(i)) = \{v\}_\omega(i) \oplus \Psi(\{v\}_\omega(i) \oplus (n_{\{v\}_\omega(i)}, a_{\{v\}_\omega(i)})) = \Psi(\{v\}_\omega(i) \oplus (n_{\{v\}_\omega(i)}, a_{\{v\}_\omega(i)})).$$

The last equality above holds because by definition $\{v\}_\omega(i) \sqsubseteq \Psi(\{v\}_\omega(i) \oplus (n_i, a_{\{v\}_\omega(i)}))$ and thus $\{v\}_\omega(i)$ can be absorbed by the latter term. Now, observing that

$$a_{\{v\}_\omega(i)} = \Psi(\{v\}_\omega(i) \oplus (n_{\{v\}_\omega(i)}, a_{\{v\}_\omega(i)}))(n_{\{v\}_\omega(i)}) = v(n_{\{v\}_\omega(i)})$$

we have $v = \Psi(\{v\}_\omega(i) \oplus (n_{\{v\}_\omega(i)}, v(n_{\{v\}_\omega(i)}))) = \Psi(\{v\}_\omega(i+1))$, which completes the induction step. □

Theorem 3.5. *Define*

$$\begin{aligned} v &:=_{X^\dagger} \Psi_X^{\varepsilon,q,\omega}(\emptyset) \\ p_i &:=_{X\to Y} \lambda x.\hat{q}(\Psi_X^{\varepsilon,q,\omega}(\{v\}_\omega(i) \oplus (n_i, x))) \end{aligned}$$

where in the second equation $i < |\mathrm{dom}(v)|$ *and* $n_i = \hat{\omega}(\{v\}(i))$*. Then for all* $0 \le i < |\mathrm{dom}(v)|$ *we have*

$$\begin{aligned} v(n_i) &= \varepsilon_{n_i} p_i \\ \hat{q}(v) &= p_i(\varepsilon_{n_i} p_i). \end{aligned} \tag{4}$$

Proof. By Lemma 3.4 we have that $v = \Psi(\{v\}_\omega(i))$ for all $0 \le i \le |\mathrm{dom}(v)|$. Using this fact, we can show, analogously to the proof of Lemma 3.4, that $n_i = \hat{\omega}(\{v\}_\omega(i)) \notin \mathrm{dom}(\{v\}_\omega(i))$ for any $0 \le i < |\mathrm{dom}(v)|$. Therefore

$$v(n_i) = \Psi(\{v\}_\omega(i))(n_i) = (\{v\}_\omega(i) \oplus \Psi(\{v\}_\omega(i) \oplus (n_i, a_{\{v\}_\omega(i)})))(n_i) = a_{\{v\}_\omega(i)}.$$

[4]Here the restriction $S_\omega(u)$ on u is merely a convenience as opposed to a necessity, allowing us to smoothly import the notation from Definition 2.1.

But by the definitions of a_u and p_i we have

$$a_{\{v\}_\omega(i)} = \varepsilon_{n_i}(\lambda x.\hat{q}(\Psi(\{v\}_\omega(i) \oplus (n_i, x)))) = \varepsilon_{n_i} p_i.$$

Put together these establish the first equation of (4). For the second we have, once more using Lemma 3.4, that for any $0 \leq i < |\mathrm{dom}(v)|$:

$$\hat{q}(v) = \hat{q}(\Psi(\{v\}_\omega(i+1))) = \hat{q}(\Psi(\{v\}_\omega(i) \oplus (n_i, v(n_i))) = p_i(v(n_i)),$$

and thus by the first equation we have $\hat{q}(v) = p_i(\varepsilon_{n_i} p_i)$. □

Corollary 3.6. *Define v and p_i in the parameters ε, q and ω as in Theorem 3.5. Let $k < |\mathrm{dom}(v)|$ be such that $n_k = \hat{\omega}(v)$. Then $f := \hat{v}$, $n := n_k$, and $p := p_k$ solve Spector's equations (1).*

Proof. First we must show that the index k is well-defined. By an easy induction similar to all those in the preceding proofs, we can show that for $0 \leq i \leq |\mathrm{dom}(v)|$ we have

$$\{v\}_\omega(i) = (n_0, v(n_0)) \oplus \ldots \oplus (n_{i-1}, v(n_{i-1}))$$

for distinct n_j, where as always $n_j = \hat{\omega}(\{v\}(j))$. In particular we have $S_\omega(v)$ and thus $v = \{v\}(|\mathrm{dom}(v)|)$. Now, since $\hat{\omega}(v) \notin \mathrm{dom}(v)$ would imply that $|\mathrm{dom}(v)| = |\mathrm{dom}(\Psi(\{v\}(|\mathrm{dom}(v)|)))| > |\mathrm{dom}(v)|$ we must have $\hat{\omega}(v) \in \mathrm{dom}(v)$ i.e. $\hat{\omega}(v) = n_k$ for some $k < |\mathrm{dom}(v)|$. The solution is now easily verified: $n = n_k = \hat{\omega}(v) = \omega(f)$ by definition, and by the equations (4) we have $f(n) = v(n_k) = \varepsilon_{n_k}(p_k) = \varepsilon_n(p)$ and $qf = \hat{q}(v) = p_k(\varepsilon_{n_k} p_k) = p(\varepsilon_n p)$. □

To summarise, so far in this section we have outlined Spector's well-known reduction of the problem of realizing the extension of the ND interpretation of classical analysis to that of solving the set of equations (1). We then recounted his standard bar recursive solution to these equations, and followed this with a novel solution using a new, symmetric variant of bar recursion. So what is the essential difference between these two approaches?

Spector's solution to the countable choice works by computing finite sequences s such that $f = \hat{s}$ forms a solution to the last pair of equations in (1) for all $n < |s|$, terminating once we have such a sequence which in addition satisfies $\omega(\hat{s}) < |s|$, thus allowing us to incorporate the first equation. This method eventually succeeds as long as we are working in a model in which Spec holds, ensuring well-foundedness of the underlying computation tree. However, while the solution given by bar recursion is elegant in its simplicity, from an algorithmic perspective it is potentially inefficient, precisely because solutions are always computed for the last two of Spector's equations for all $n < |s|$ whereas we only need these equations to hold for $n = \omega(\hat{s})$.

Our method of constructing solutions to Spector's equations uses a new recursor which constructs finite partial functions u such that $f = \hat{u}$ forms a solution to the last pair of equations for all $n \in \mathrm{dom}(u)$, where the set $\mathrm{dom}(u)$ is guided by the parameter ω. This means that, in stark contrast to Spector's method, we do not necessarily need to have computed solutions for the last equations for every n in some initial segment of $\mathbb{N}$, but only for certain values. While our solution is somewhat more complicated to verify, and in particular is based on a form of recursion for which it is seemingly more difficult to prove termination (cf. Section 2), from a purely practical perspective it is possible that it gives rise to much more efficient programs.

Note that when we talk of efficiency, we are intuitively referring to intensional aspects of the recursors - i.e. algorithms for them, rather than the recursors themselves. We could make this more formal by considering a rewrite system defined by the recursive equations, and then the recursors could be compared more precisely by counting the length of reduction sequences that arise when computing realizers.

However, because we are ultimately interested in how the recursors perform when actually implemented on a computer, in the next section we take a slightly less abstract approach and examine the performance of the Haskell programs defined by the recursors. We present a short and informal case study in order to provide a concrete illustration of the differences between the two methods of program extraction, our aim being to highlight that in *practice* realizers based on symmetric bar recursion compare favourably to the traditional Spector bar recursion.

4 Case study: No injection from $\mathbb{N} \to \mathbb{N}$ to $\mathbb{N}$

We illustrate the preceding theoretical results with a program extraction from the proof that there is no injection from the function space $\mathbb{N} \to \mathbb{N}$ to the natural numbers $\mathbb{N}$. This theorem can be formalized as a Π_2-statement in the language of $\mathsf{E\text{-}PA}^\omega$, and moreover its standard proof by a diagonal argument can be formalized using an instance of $\mathsf{AC}_{\mathbb{N},\mathbb{N}\to\mathbb{N}}$.

This example was originally used by the first author in [18] in order to demonstrate the extraction of programs from proofs using Spector's bar recursor, and is a good candidate for a case study as it is relatively straightforward without being completely trivial.

Theorem 4.1. $\mathsf{E\text{-}PA}^\omega + \mathsf{AC}_{\mathbb{N},\mathbb{N}\to\mathbb{N}} \vdash \forall H\colon \mathbb{N}^{\mathbb{N}} \to \mathbb{N}\, \exists \alpha, \beta\colon \mathbb{N} \to \mathbb{N}\, \exists i : \mathbb{N}(\alpha i \neq \beta i \wedge H\alpha = H\beta)$.

Classical Proof. As a simple case of the law of excluded middle (also known as the drinker's paradox) we have

$$\forall n^{\mathbb{N}} \exists \alpha^{\mathbb{N}\to\mathbb{N}} (\exists \beta (H\beta = n) \to H\alpha = n). \tag{5}$$

Applying $\mathsf{AC}_{\mathbb{N},\mathbb{N}\to\mathbb{N}}$ to (5) yields a functional $f\colon \mathbb{N} \to \mathbb{N}^{\mathbb{N}}$ satisfying

$$\forall n (\exists \beta (H\beta = n) \to H(f(n)) = n). \tag{6}$$

The map f produces for each n a function $f(n)\colon \mathbb{N} \to \mathbb{N}$ such that whenever n is in the range of H, $f(n)$ maps to n. Now, define $\alpha_H := \lambda n.f(n)(n) + 1$ and let $i_H := H(\alpha_H)$ and $\beta_H := f(i_H)$. Then since i_H is in the range of H, by (6) we must have $H(\beta_H) = H(f(i_H)) = i_H = H(\alpha_H)$. But $\alpha_H \neq f(i_H) = \beta_H$. □

It is an intriguing consequence of Spector's ND interpretation of classical analysis that we are *a priori* guaranteed to be able to convert the classical diagonal argument above into a semi-intuitionistic proof, and directly construct (using bar recursion) explicit witnesses for α_H, β_H and i_H as a function of H. Moreover, Spector's reduction of the ND interpretation of analysis to the countable choice problem demonstrates that a realizer for Theorem 4.1 can be constructed primitive recursively in an *arbitrary* solution to the equations (1). In particular, we can replace Spector's bar recursion with an instance of symmetric bar recursion to give an alternative procedure for refuting injectiveness.

Proposition 4.2. *Any computable solution to Spector's equations allows us to effectively extract witnesses for α, β and i in Theorem 4.1.*

Proof. A solution to Spector's equations acts as a computational interpretation of an instance of $\mathsf{AC}_{\mathbb{N}}$. We must simply produce suitable parameters for these equations which correspond to the particular instance of $\mathsf{AC}_{\mathbb{N},\mathbb{N}\to\mathbb{N}}$ used in the classical proof. First, we give a computation interpretation to the initial instance of law of excluded middle via the term $\varepsilon\colon \mathbb{N} \to (\mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}) \to \mathbb{N}^{\mathbb{N}}$ given by

$$\varepsilon_n(p^{\mathbb{N}^{\mathbb{N}}\to\mathbb{N}^{\mathbb{N}}}) :=_{\mathbb{N}^{\mathbb{N}}} \begin{cases} \mathbf{0}_{\mathbb{N}\to\mathbb{N}} & \text{if } H(p(\mathbf{0})) \neq n \\ p(\mathbf{0}) & \text{if } H(p(\mathbf{0})) = n. \end{cases} \tag{7}$$

It is easy to verify that ε satisfies

$$\forall n, p \left(H(p(\varepsilon_n p)) = n \to H(\varepsilon_n p) = n\right), \tag{8}$$

which is just the ND interpretation of (5). Now, a computable solution to Spector's equations allows us to effectively construct an f in $q\colon (\mathbb{N} \to \mathbb{N}^{\mathbb{N}}) \to \mathbb{N}^{\mathbb{N}}$ and $\omega\colon (\mathbb{N} \to \mathbb{N}^{\mathbb{N}}) \to \mathbb{N}$ satisfying

$$H(q(f)) = \omega(f) \to H(f(\omega f)) = \omega(f) \tag{9}$$

using (8) and Spector's equations (1). Therefore, defining $q(g) := \lambda n.g(n)(n) + 1$ and $\omega(g) := H(q(g))$, the premise of (9) holds by definition and hence we obtain $H(f(\omega f)) = \omega(f)$. Finally, setting $\alpha_H := q(f)$ and $\beta_H := f(\omega f)$ we have $H\beta_H = H\alpha_H$, but α_H and β_H differ at $i_H := \omega(f)$. □

In a general manner of speaking, the reason one is are able to convert the classical proof of Theorem 4.1 into a construction which computes α and β for any given H is that the solution of Spector's equations will typically only work in a subset of the full set-theoretic type structure. Solutions can be obtained, for instance, if either continuity or majorizability is assumed (cf. [9, 23]), although Proposition 4.2 provides a solution that is independent of the particular model one has in mind.

However, the exact computational process in calculating these witnesses will depend on our chosen solution of Spector's equations. We now briefly analyse the program which arises from choosing our symmetric bar recursive solution in place of Spector's original bar recursion.

4.1 Numerical performance on sample input

We have implemented both Spector's and our new variant of bar recursion in Haskell[5]. We ran several tests in which we used both variants to compute counterexample functions α_H and β_H as described in Proposition 4.2 for various concrete choices of H. To make things clearer, recall that

$$\alpha_H = q(f) \qquad\qquad \beta_H = f(H(q(f)))$$

where f is some solution to Spector's equations (1) in ε, q and ω, which are all defined as in the proof of the Proposition. By Theorems 3.1 and 3.5 our counterexamples for the two forms of bar recursion are given as follows:

$$\begin{array}{rlll} \text{Spector:} & \alpha_H = \hat{q}(\Phi^{\varepsilon,q,\omega}(\langle\rangle)) & \beta_H = \Phi^{\varepsilon,q,\omega}(\langle\rangle)(i_H) & i_H = H(\hat{q}(\Phi^{\varepsilon,q,\omega}(\langle\rangle))) \\ \text{Symmetric:} & \alpha_H = \hat{q}(\Psi^{\varepsilon,q,\omega}(\emptyset)) & \beta_H = \Psi^{\varepsilon,q,\omega}(\emptyset)(i_H) & i_H = H(\hat{q}(\Psi^{\varepsilon,q,\omega}(\emptyset))). \end{array}$$

For each instance of H we calculated

(a) the domain size of the finite approximations $\Phi(\langle\rangle)\colon (\mathbb{N}^{\mathbb{N}})^*$ and $\Psi(\emptyset)\colon (\mathbb{N}^{\mathbb{N}})^\dagger$, and

(b) the number of recursive calls triggered when computing i_H together with the values of α_H and β_H up to this point.

Of course, we could have chosen other benchmarks by which to compare the realizers, although we regard this as being fairly representative of how we might want to use the realizer in practice. In any case our only, modest aim in this section is to give an informal comparison between the two methods of program extraction.

In the vast majority of natural cases we have considered the procedure based on symmetric bar recursion outperformed that based on Spector's bar recursion by a considerable margin. The interested reader is encouraged to try their own examples using our source code to witness this for themselves. However, we sketch a few representative examples here.

Example 1. Take the family of functionals H_n defined by

$$H_n(\gamma^{\mathbb{N}\to\mathbb{N}}) = \Pi_{i=0}^{n-1}(1+\gamma i).$$

The table below indicates the domain sizes of $\Phi(\langle\rangle)$ and $\Psi(\emptyset)$, and the number of recursive calls needed to calculate i_H and $[\alpha_H](i_H+1)$ and $[\beta_H](i_H+1)$, for $n \in \{4,5,6\}$.

	Spector (domain size / # recursive calls)	Symmetric (domain size / # recursive calls)
$n=4$	17 / 1140	1 / 12
$n=5$	33 / 4650	1 / 12
$n=6$	65 / 19154	1 / 12

This disparity in performance can be intuitively understood by computing on paper what each recursor does. First, take the symmetric recursor. One can show that

$$\hat{\omega}(\emptyset) = H_n(q(\mathbf{0}_{\mathbb{N}\to\mathbb{N}\to\mathbb{N}})) = H_n(\lambda i.1) = \Pi_{i=0}^{n-1}(1+1) = 2^n$$

and thus $\Psi(\emptyset) = \Psi((2^n, a_\emptyset))$ where

$$a_\emptyset := \varepsilon_{2^n}(\lambda x\,.\,\hat{q}(\Psi((2^n,x)))) = \begin{cases} \mathbf{0}_{\mathbb{N}\to\mathbb{N}} & \text{if } H_n(\hat{q}(\Psi((2^n,\mathbf{0})))) \neq 2^n \\ \hat{q}(\Psi((2^n,\mathbf{0}))) & \text{otherwise.} \end{cases}$$

But since $\hat{\omega}((2^n,\mathbf{0})) = \hat{\omega}(\emptyset) = 2^n \in \mathrm{dom}((2^n,\mathbf{0})) = \{2^n\}$ we see that $\Psi((2^n,\mathbf{0})) = (2^n,\mathbf{0})$, and therefore $\hat{q}(\Psi((2^n,\mathbf{0}))) = \lambda k.1$ and thus $a_\emptyset = \lambda k.1$. Now, it is easy to show that we also have

$$\hat{\omega}((2^n,\lambda k.1)) = \Pi_{i=0}^{n-1}(1+1) = 2^n$$

[5] `http://www.eecs.qmul.ac.uk/~pbo/code/bar/`

and thus the recursion terminates, yielding $\Psi(\emptyset) = \Psi((2^n, \lambda k.1)) = (2^n, \lambda k.1)$. Our counterexample are then

$$\alpha_{H_n} = \hat{q}((2^n, \lambda k.1)) = \lambda k.\text{ if } k = 2^n \text{ then 2 else 1}$$
$$\beta_{H_n} = \Psi(\emptyset)(H_n(\alpha_{H_n})) = \Psi(\emptyset)(2^n) = \lambda k.1$$

and it is easy to verify that these counterexamples indeed work. Moreover, in order to compute them our symmetric bar recursor only needed to produce a finite partial function with one element, resulting in a very quick algorithm.

In contrast, take Spector's bar recursor. Entirely analogously to before we have

$$\hat{\omega}(\langle\rangle) = H_n(q(\mathbf{0}_{\mathbb{N}\to\mathbb{N}\to\mathbb{N}})) = 2^n$$

but now since the recursor is forced to compute sequentially we have $\Phi(\langle\rangle) = \Phi(\langle a_{\langle\rangle}\rangle)$ where

$$a_{\langle\rangle} := \varepsilon_0(\lambda x\,.\,\hat{q}(\Phi(\langle x\rangle))) = \begin{cases} \mathbf{0}_{\mathbb{N}\to\mathbb{N}} & \text{if } H_n(\hat{q}(\Phi(\langle\mathbf{0}\rangle))) \neq 0 \\ \hat{q}(\Phi(\langle\mathbf{0}\rangle)) \neq 0 & \text{otherwise.} \end{cases}$$

Since $\hat{\omega}\langle\mathbf{0}\rangle = \hat{\omega}(\langle\rangle) = 2^n > |\langle\mathbf{0}\rangle|$ we have that $\Phi(\langle\mathbf{0}\rangle) = \langle\mathbf{0}\rangle \oplus \Phi(\langle\mathbf{0}, a_{\langle\mathbf{0}\rangle}\rangle)$ and we must continue by computing $a_{\langle\mathbf{0}\rangle}$. Expanding the definition of $a_{\langle\mathbf{0}\rangle}$ in terms of ε_1 analogously requires a further recursive call to $\Phi(\langle\mathbf{0},\mathbf{0}\rangle)$, and since once more $\hat{\omega}\langle\mathbf{0},\mathbf{0}\rangle = \hat{\omega}(\langle\rangle) = 2^n > |\langle\mathbf{0},\mathbf{0}\rangle|$ we have $\Phi(\langle\mathbf{0},\mathbf{0}\rangle) = \langle\mathbf{0},\mathbf{0}\rangle \oplus \Phi(\langle\mathbf{0},\mathbf{0}, a_{\langle\mathbf{0},\mathbf{0}\rangle}\rangle)$ and so on. In fact, our recursor continues to make nested recursive calls in this fashion until it reaches the list $\langle\mathbf{0},\ldots,\mathbf{0}\rangle$ of size 2^n, in which case one can compute that $\Phi(\langle\mathbf{0}\ldots\mathbf{0}\rangle) = \langle\mathbf{0},\ldots,\mathbf{0},\lambda k.1\rangle$. Backtracking one finally obtains

$$\Phi(\langle\rangle) = \Phi(\langle\mathbf{0}\rangle) = \ldots = \langle\underbrace{\mathbf{0},\ldots,\mathbf{0}}_{2^n \text{ times}},\lambda k.1\rangle.$$

Our counterexamples are then

$$\alpha_{H_n} = \hat{q}(\langle\mathbf{0},\ldots,\mathbf{0},\lambda k.1\rangle) = \lambda k.\text{ if } k = 2^n \text{ then 2 else 1}$$
$$\beta_{H_n} = \Psi(\emptyset)(H_n(\alpha_{H_n})) = \Psi(\emptyset)(2^n) = \lambda k.1$$

which are exactly the same as those produced by the symmetric bar recursor. However, the algorithm induced by the Spector recursor was forced to carry out a lengthy backtracking procedure along sequences of length up to 2^{n+1}, resulting in a much more complex computation.

If we adjust H_n to make it more complex still, for example

$$H_n(\gamma) = \Pi_{i=0}^{n-1}(1+i)^{1+\gamma i},$$

the disparity is even more extreme:

	Spector	Symmetric
$n = 3$	577 /2350	1 / 12
$n = 4$	577 /365700	1 / 12

Example 2. Now suppose $H_n(\gamma)$ searches for a least point such that $\gamma i < \gamma(i+1)$:

$$H_n(\gamma) = \text{least } i \leq n \text{ such that } \gamma i < \gamma(i+1), \text{ else } n \text{ if none exist.}$$

The corresponding data for $n \in \{3,4,5\}$ is now:

	Spector	Symmetric
$n = 3$	4 / 316	4 / 52
$n = 4$	5 / 688	5 / 64
$n = 5$	6 / 1444	6 / 76

In this case, both recursors terminate once they have computed a domain of size n, but Spector's recursor takes much longer. Once more, this behaviour has an intuitive explanation. Let us take $n = 3$ for simplicity, and define functions $\mathbb{N} \to \mathbb{N}$ via the following notational convention, whereby

$$\gamma := [x_0, x_1, \ldots, x_{k-1}]$$

means that $\gamma i = x_i$ for $i < k$ and $\gamma i = 1$ otherwise. Let's first look at the symmetric recursor. We have

$$\hat{\omega}(\emptyset) = H_3(q(\mathbf{0})) = H_3([1,1,1,1]) = 3$$

since H_3 is forced to return the 'else' clause, and therefore $\Psi(\emptyset) = \Psi((3, \gamma_0))$ where we write $\gamma_0 := a_\emptyset = \varepsilon_3(\lambda x \,.\, \hat{q}(\Psi((3,x))))$. The functional ε_3 be expanded just as in the previous section, and analogously to there we can check that $\hat{q}(\Psi((3,\mathbf{0}))) = \hat{q}((3,\mathbf{0})) = [1,1,1,1]$ and thus $\gamma_0 = [1,1,1,1]$. Now we have

$$\hat{\omega}((3,\gamma_0)) = H_3(\hat{q}((3,\gamma_0))) = H_3([1,1,1,2]) = 2.$$

Thus $\Psi((3,\gamma_0)) = \Psi((3,\gamma_0) \oplus (2,\gamma_1))$ where $\gamma_1 := a_{(3,f_0)} = \varepsilon_2(\lambda x \,.\, \hat{q}(\Psi((3,\gamma_0) \oplus (2,x))))$. Now again we can verify that $\hat{q}(\Psi((3,\gamma_0) \oplus (2,\mathbf{0}))) = \hat{q}((3,\gamma_0) \oplus (2,\mathbf{0})) = [1,1,1,2]$ and thus $f_1 = [1,1,1,2]$. This process continues until we obtain

$$\Psi(\emptyset) = (0,[1,2,2,2]) \oplus (1,[1,1,2,2]) \oplus (2,[1,1,1,2]) \oplus (3,[1,1,1,1]),$$

from which we read off $\alpha_{H_3} = \hat{q}(\Psi(\emptyset)) = [2,2,2,2]$ and $\beta_{H_3} = \Psi(\emptyset)(3) = [1,1,1,1]$, and it is clear that these are valid counterexamples.

Entirely analogously, for general n we obtain

$$\alpha_{H_n} = [\underbrace{2,2,\ldots,2}_{n+1 \text{ times}}] \quad \text{and} \quad \beta_{H_n} = [\underbrace{1,1,\ldots,1}_{n+1 \text{ times}}]$$

In the case of Spector's bar recursion, it is clear that since we always have $\omega(f) = H_n(q(f)) \leq n$ then the recursor will output a finite sequence of length at most n, and so the domain of the bar-recursive output is no bigger that that of symmetric bar recursion. However, as in the first example the computation itself involves a much more intricate backtracking procedure, and this time also yields a slightly different solution. By a somewhat tedious calculation (which the reader can try to reproduce themselves if they want convincing of the complexity of Spector's recursion!) one obtains for e.g. $n = 3$

$$\Phi(\langle\rangle) = \langle [1,2,1,1], [2,1,2,1], [2,2,1,2], [2,2,2,1] \rangle$$

yielding solutions $\alpha_{H_n} = [2,2,2,2]$ and $\beta_{H_n} = [2,2,2,1]$ and analogously for general n:

$$\alpha_{H_n} = [\underbrace{2,2,\ldots,2}_{n+1 \text{ times}}] \quad \text{and} \quad \beta_{H_n} = [\underbrace{2,2,\ldots,2}_{n \text{ times}},1]$$

However, due again to its insistence of making recursive calls sequentially, Spector's recursor takes much longer to compute these solutions, as is clear from the table.

Summary. Naturally, this section remains very informal given that we have only provided a couple of examples to illustrate the difference between programs obtained using our recursor and those obtained via the traditional Spector bar recursor. Moreover, it is not the case that the symmetric recursor *always* produces a more efficient algorithm than the Spector – it is not too hard to come up with a somewhat contrived H_n for which a sequential bar recursion is clearly better. Take, for example

$$H_n(\gamma) = \begin{cases} \text{greatest } i \leq n(\gamma(i) = 1) \text{ if it exists, else } n & \text{if } \gamma(0) = \gamma(1) = 2 \\ 0 & \text{if } \gamma(0) = 1 \wedge \gamma(1) = 2 \text{ or } \gamma(0) = 2 \wedge \gamma(1) = 1 \\ 1 & \text{otherwise.} \end{cases}$$

Here the sequential computation associated with Φ means that the first clause in the case distinction is never triggered, so that Φ always returns a sequence of length 2. On the other hand, Ψ ends up with a finite partial function of size n, and so its cost is proportional to n.

Nevertheless, the fact that such a H exists does not necessarily detract from our symmetric bar recursion being a useful alternative to Spector's bar recursion in practice. In particular, when *using* our realizer as, for instance, a building block for a more complex realizing term arising from a classical proof that uses Theorem 4.1 as a lemma, it is reasonable to assume that H will take the form of a fairly natural recursive function, and the authors conjecture that in many such cases symmetric bar recursion will drastically outperform Spector's bar recursion.

On top of this, due to the fact that it often avoids unnecessary backtracking, we believe that our symmetric recursor will often give rise to programs that are more natural and easier to understand from an *algorithmic* perspective. In particular, we conjecture that there are close links between the symmetric bar recursive interpretation of choice over sequences law of excluded middle for Σ_1-formulas (of which the instance of $\mathsf{AC}_{\mathbb{N}}$ here is an example) and the learning procedures for $\mathsf{PA} + \mathsf{EM}_1$ described in [1].

However, we leave any further analysis to future work. For now, our main achievement has been to devise an interesting symmetric alternative to Spector's bar recursion, which appears to us more sensible and natural for the purpose of program extraction, and in a small number of test cases drastically outperforms the latter.

5 Equivalence of BR and sBR

In this section we prove that the recursion schemata BR and sBR are actually primitive recursively equivalent. This is the most technical part of the paper, but is entirely self-contained and is not at all necessary in order to understand the preceding sections. The most difficult direction – the definability of sBR from BR – can be carried out in $\mathsf{E\text{-}HA}^\omega + \mathsf{sBI}$ and hence (by Theorem 2.7) in $\mathsf{E\text{-}PA}^\omega + \mathsf{DC}$, and thus as an immediate consequence we prove that sBI exists in any model of $\mathsf{E\text{-}PA}^\omega + \mathsf{DC}$ which also validates BR. In particular, sBI exists in both the Kleene/Kreisel total continuous functionals (as stated earlier in Theorem 2.11) but also the strongly majorizable functionals.

5.1 BR is definable from sBR

Defining BR from sBR is somewhat easy. The basic idea is that $\mathsf{BR}_{X,R}$ can be defined from a single instance of $\mathsf{sBR}_{X\times\mathbb{B},R}$ of (essentially) the same type, in which the symmetric control functional $\tilde{\omega}$ is designed to be 'stubborn' and always search for the least undefined point to update, thereby simulating Spector's bar recursion. When we view the finite sequence $s\colon X^*$ as a finite partial function $s\colon X^\dagger$, we have that the domain of s is an initial segment of $\mathbb{N}$. So we can use $\mathsf{sBR}_{X\times\mathbb{B},R}$ with a control function $\tilde{\omega}$ that looks for the smallest number which is not in that initial segment, therefore mimicking the behaviour of BR. It might look like this requires an unbounded search, but in fact, the original control function ω provides a bound for the search.

Theorem 5.1. *$\mathsf{BR}_{X,R}$ is primitive recursively definable from $\mathsf{sBR}_{X\times\mathbb{B},R}$, provably in $\mathsf{E\text{-}HA}^\omega + \mathsf{sBR}$.*

Proof. Suppose we are given parameters $\phi\colon X^* \to (X \to R) \to R$, $b\colon X^* \to R$ and $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$ for $\mathsf{BR}_{X,R}$. Then there is a term $\Phi^{\phi,b,\omega}$ primitive recursive in $\mathsf{sBR}_{X\times\mathbb{B},R}$ that satisfies the defining equation of $\mathsf{BR}_{X,R}^{\phi,b,\omega}$. Recall that we identify $\mathbb{B}$ with the set $\{0,1\}$, taking $\mathbf{0}_{\mathbb{B}} := 0$, and thus $\mathbf{0}_{X\times\mathbb{B}} \equiv \langle \mathbf{0}_X, 0\rangle$. Define the map $\eta\colon X^* \to (X\times\mathbb{B})^\dagger$ by

$$(\eta s)(n) := \begin{cases} \langle s_n, 1\rangle & \text{if } n < |s| \\ \text{undefined} & \text{otherwise,} \end{cases}$$

so that $\mathrm{dom}(\eta s) = \{0, 1, \ldots, |s|-1\}$; and conversely the map $\eta'\colon (X\times\mathbb{B})^\dagger \to X^*$ by $|\eta' u| = N+1$ where N is the maximum element of $\mathrm{dom}(u)$, and

$$(\eta' u)_i := \begin{cases} \pi_0(u(i)) & \text{if } i \in \mathrm{dom}(u) \\ \mathbf{0}_X & \text{otherwise} \end{cases}$$

where $\pi_0\colon X\times\mathbb{B} \to X$ is the first projection. Note that $\eta'\eta s = s$ for all $s\colon X^*$. Now, define parameters

$\tilde{\phi}$, $\tilde{b}$ and $\tilde{\omega}$ for $\mathsf{sBR}_{X\times\mathbb{B},R}$ by

$$\begin{aligned}\tilde{\omega}(\alpha^{(X\times\mathbb{B})^{\mathbb{N}}}) &:\overset{\mathbb{N}}{=} \mu i \leq \omega(\pi_0\alpha)((\pi_1\alpha)(i) =_{\mathbb{B}} 0)\\ \tilde{b}(u^{(X\times\mathbb{B})^\dagger}) &:\overset{R}{=} b(\eta' u)\\ \tilde{\phi}_u(p^{X\times\mathbb{B}\to R}) &:\overset{R}{=} \phi_{[\pi_0\hat{u}](\tilde{\omega}(\hat{u}))}(\lambda x^X.p(\langle x,1\rangle))\end{aligned}$$

where $\pi_i\alpha$ denotes the projection of the sequence α i.e. $(\pi_i\alpha)(n) := \pi_i(\alpha(n))$. We claim that

$$\Delta^{\phi,b,\omega}(s) := \mathsf{sBR}^{\tilde{\phi},\tilde{b},\tilde{\omega}}_{X\times\mathbb{B},R}(\eta s)$$

satisfies the defining equation of $\mathsf{BR}^{\phi,b,\omega}_{X,R}(s)$. To prove this, first note that

(*i*) $\pi_0(\widehat{\eta s}) =_{X^{\mathbb{N}}} \hat{s}$, and

(*ii*) $\pi_1(\widehat{\eta s})(i) =_{\mathbb{B}} 1$, for $i < |s|$ and 0 otherwise,

follow directly from the definition of η and the fact that $\mathbf{0}_{X\times\mathbb{B}} = \langle \mathbf{0}_X, 0\rangle$. Therefore

$$(iii)\ \ \tilde{\omega}(\widehat{\eta s}) = \mu i \leq \omega(\pi_0(\widehat{\eta s})))(\pi_1(\widehat{\eta s})(i) =_{\mathbb{B}} 0) \overset{(i)}{=} \mu i \leq \omega(\hat{s})(\pi_1(\widehat{\eta s})(i) = 0) \overset{(ii)}{=} \begin{cases}\omega(\hat{s}) & \text{if } \omega(\hat{s}) < |s|\\ |s| & \text{if } \omega(\hat{s}) \geq |s|.\end{cases}$$

Since $\mathrm{dom}(\eta s) = \{0,\ldots,|s|-1\}$ point (*iii*) above implies the equivalence

(*iv*) $\omega(\hat{s}) < |s| \ \Leftrightarrow\ \tilde{\omega}(\widehat{\eta s}) \in \mathrm{dom}(\eta s)$.

Therefore, if $\omega(\hat{s}) < |s|$ then $\tilde{\omega}(\widehat{\eta s}) \in \mathrm{dom}(\eta s)$ and hence

$$\Delta(s) = \mathsf{sBR}^{\tilde{\phi},\tilde{b},\tilde{\omega}}_{X\times\mathbb{B},R}(\eta s) = \tilde{b}(\eta s) = b(\eta'\eta s) = b(s).$$

If $\omega(\hat{s}) \geq |s|$ then $\tilde{\omega}(\widehat{\eta s}) = |s| \notin \mathrm{dom}(\eta s)$ and hence

$$\begin{aligned}\Delta(s) &= \mathsf{sBR}^{\tilde{\phi},\tilde{b},\tilde{\omega}}_{X\times\mathbb{B},R}(\eta s)\\ &= \tilde{\phi}_{\eta s}(\lambda\langle x,b\rangle^{X\times\mathbb{B}}.\,\mathsf{sBR}(\eta s \oplus (|s|,\langle x,b\rangle)))\\ &= \phi_{[\pi_0(\widehat{\eta s})](\tilde{\omega}(\widehat{\eta s}))}(\lambda x\,.\,\mathsf{sBR}(\eta s \oplus (|s|,\langle x,1\rangle)))\\ &\overset{(i)}{=} \phi_{[\hat{s}](|s|)}(\lambda x\,.\,\mathsf{sBR}(\eta s \oplus (|s|,\langle x,1\rangle)))\\ &= \phi_s(\lambda x\,.\,\mathsf{sBR}(\eta(s*x)))\\ &= \phi_s(\lambda x\,.\,\Delta(s*x))\end{aligned}$$

where for the penultimate equality one easily verifies that $\eta s \oplus (|s|,\langle x,1\rangle) = \eta(s*x)$. □

5.2 sBR is definable from BR

A similar idea, however, does not seem to work in the opposite direction, since Spector's bar recursion is inherently less flexible than symmetric bar recursion. Instead, to define $\mathsf{sBR}_{X,R}$ we resort to an instance of Spector's bar recursion of a strictly higher type, and the resulting construction is somewhat more intricate.

Theorem 5.2. *$\mathsf{sBR}_{X,R}$ is primitive recursively definable from $\mathsf{BR}_{X^\dagger\times(X^\dagger\to(X\to R)),R}$, provably in $\mathsf{E\text{-}HA}^\omega + \mathsf{sBI} + \mathsf{BR}$.*

We break up the proof of Theorem 5.2 into several steps. The basic idea behind our construction is as follows: a finite partial state u in the computation of sBR is represented by a finite sequence of pairs s_u in our instance of BR. For each n, the value $s_u(n)$ will have type $X^\dagger \times (X^\dagger \to (X \to R))$. If $n \in \mathrm{dom}(u)$ then the first component of $s_u(n)$ contains a smaller "state" $u' \sqsubseteq u$ that was present when point n was updated, and if $n \notin \mathrm{dom}(u)$ then the second component of $s_u(n)$ contains a continuation that allows us to make bar recursive calls on updates of the state at any point in the future allowing us to simulate the behaviour of sBR. First, we need some definitions.

Definition 5.3. Suppose $u, v\colon X^\dagger$ and $x\colon X$. Then $[u](n) * x \oplus v\colon X^\dagger$ denotes the finite partial function given by

$$([u](n) * x \oplus v)(i) = \begin{cases} u(i) & \text{if } i < n \\ x & \text{if } i = n \\ v(i) & \text{if } i > n. \end{cases}$$

So $\mathrm{dom}([u](n) * x \oplus v) = (\mathrm{dom}(u)\backslash\{n, \ldots\}) \cup \{n\} \cup (\mathrm{dom}(v)\backslash\{0, \ldots, n\})$. Next, let us introduce the type abbreviation $Y :\equiv X^\dagger \times (X^\dagger \to (X \to R))$, and define the 'diagonal' functional $d\colon Y^* \to X^\dagger$ by

$$d(s)(j) := \begin{cases} (\pi_0 s_i)(j) & \text{for least } i \leq |s| \text{ such that } i < j \text{ and } j \in \mathrm{dom}(\pi_0 s_i) \\ \text{undefined} & \text{otherwise.} \end{cases}$$

The function d returns a particular kind of 'union' of the finite partial functions $\pi_0 s_i$, where in the event that the $\pi_0 s_i$ are defined at j for more than one index $i \leq j$, the value at the least index is chosen. Because s is a finite sequence the resulting partial function $d(s)$ must also have finite domain. Similarly, define an infinitary diagonal function $d^\infty\colon Y^{\mathbb{N}} \to X^{\mathbb{N}}$ by

$$d^\infty(\alpha)(j) := \begin{cases} (\pi_0 \alpha_i)(j) & \text{for least } i \leq j \text{ such that } j \in \mathrm{dom}(\pi_0 \alpha_i) \\ \mathbf{0}_X & \text{otherwise.} \end{cases}$$

This function d^∞ returns a similar union of the infinite sequence of partial functions $\pi_0 \alpha_i$, returning a partial function with potentially infinite domain, and then embedding this partial function in $X^{\mathbb{N}}$ by assigning the canonical value $\mathbf{0}_X$ to undefined elements.

The main step in our proof of Theorem 5.2 will be to show that BR defines a slightly altered form of sBR, which only accepts ω-threads as input. As we show in Lemma 5.12 this restriction is inessential – however, it makes the verification slightly easier to work with and hence we adopt this variant for now.

Lemma 5.4. *Let $\Theta_{X,R}$ be the following variant of sBR,*

$$\Theta^{\phi,b,\omega}(u^{X^\dagger}) = \begin{cases} \mathbf{0}_R & \text{if } \neg S_\omega(u) \\ b(u) & \text{if } \omega(\hat{u}) \in \mathrm{dom}(u) \\ \phi_u(\lambda x.\Theta^{\phi,b,\omega}(u \oplus (\omega(\hat{u}), x))) & \text{otherwise} \end{cases} \tag{10}$$

where the parameters have the same type as those for sBR, namely $\phi\colon X^\dagger \to (X \to R) \to R$, $b\colon X^\dagger \to R$ and $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$. Then $\Theta_{X,R}$ is primitive recursively definable from Spector's bar recursion $\mathsf{BR}_{Y,R}$ for $Y :\equiv X^\dagger \times (X^\dagger \to (X \to R))$, provably in $\mathsf{E\text{-}HA}^\omega + \mathsf{BR}$.

Proof. Suppose we are given parameters ϕ, b and ω for Θ. We define parameters $\tilde{\phi}$, $\tilde{b}$ and $\tilde{\omega}$ for $\mathsf{BR}_{Y,R}$ in terms of ϕ, b and ω. We begin with $\tilde{\phi}$, which is given by

$$\tilde{\phi}_{s^{Y^*}}(p^{Y \to R}) :\overset{R}{=} \begin{cases} p(\langle d(s), \mathbf{0}_{X^\dagger \to (X \to R)} \rangle) & \text{if } |s| \in \mathrm{dom}(d(s)) \\ p(\langle d(s), \lambda v^{X^\dagger}, x^X.p(\langle [d(s)](|s|) * x \oplus v, \mathbf{0}_{X^\dagger \to (X \to R)} \rangle) \rangle) & \text{otherwise.} \end{cases}$$

As will become clear below, p plays the role of a continuation: If $d(s)$ is not defined at $|s|$, it initiates a nested recursive call to variants of $d(s)$ of the form $[d(s)](|s|) * x \oplus v$, which are identical to $d(s)$ for all arguments $i < |s|$ but are now defined at point $|s|$ with value x. For the parameter $\tilde{b}$, define

$$\tilde{b}(s^{Y^*}) :\overset{R}{=} \begin{cases} b(d(s)) & \text{if } \hat{\omega}(d(s)) \in \mathrm{dom}(d(s)) \\ \phi_{d(s)}(\pi_1(\hat{s}_{\hat{\omega}(d(s))})(d(s))) & \text{otherwise.} \end{cases}$$

Note that this is well typed since $\hat{s}\colon Y^{\mathbb{N}}$, and thus $\hat{s}_{\hat{\omega}(d(s))}\colon Y$ and $\pi_1(\hat{s}_{\hat{\omega}(d(s))})\colon X^\dagger \to (X \to R)$, which implies that $\pi_1(\hat{s}_{\hat{\omega}(d(s))})(d(s))\colon X \to R$. For the final parameter, let

$$\tilde{\omega}(\alpha^{Y^{\mathbb{N}}}) :\overset{\mathbb{N}}{=} \omega(d^\infty(\alpha)).$$

We now define a sequence of finite sequences $s_{u,i}\colon Y^*$ for $u\colon X^\dagger$, primitive recursively in $\mathsf{BR}^{\tilde{\phi},\tilde{b},\tilde{\omega}}_{Y,R}$, as

$$s_{u,0} := \langle\rangle$$
$$s_{u,i+1} := \begin{cases} [s_{u,i}](n_i) * \langle u_{i+1}, \mathbf{0}_{X^\dagger\to(X\to R)}\rangle & \text{if } n_i < |s_{u,i}| \\ s_{u,i} * \langle d(s_{u,i}), f^i_{|s_{u,i}|}\rangle * \ldots * \langle d(s_{u,i}), f^i_{n_i-1}\rangle * \langle u_{i+1}, \mathbf{0}\rangle & \text{otherwise} \end{cases}$$

where we use the abbreviations $u_i := \{u\}_\omega(i)$ and $n_i := \hat{\omega}(u_i)$, and the functions f^i_n are defined using course-of-values recursion as

$$f^i_n := \begin{cases} \lambda w^{X^\dagger}, x^X.\mathsf{BR}^{\tilde{\phi},\tilde{b},\tilde{\omega}}_{Y,R}([s_{u,i+1}](n) * \langle [d(s_{u,i})](n) * x \oplus w, \mathbf{0}\rangle) & \text{if } n \notin \mathrm{dom}(d(s_{u,i})) \\ \mathbf{0}_{X^\dagger\to(X\to R)} & \text{otherwise.} \end{cases} \tag{11}$$

While the definition of $s_{u,i+1}$ may seem circular, in that f^i_n uses $s_{u,i+1}$, it is well-defined by course-of-values recursion along the length of $s_{u,i+1}$ because $f^i_n = \pi_1(s_{u,i+1}(n))$, and to define this we only require knowledge of $[s_{u,i+1}](n)$ i.e. $s_{u,i+1}(m)$ for $m < n$, and for the base case $f^i_{|s_{u,i}|}$ is defined in terms of $[s_{u,i+1}](|s_{u,i}|) = s_{u,i}$.
Now, let $s_u := s_{u,|\mathrm{dom}(u)|}$, so that $s_u\colon Y^*$. It is easy to see that $|s_{u,i+1}| = n_i + 1$ for arbitrary i, which means that in particular $|s_u| = n_{|\mathrm{dom}(u)|-1} + 1$ whenever $|\mathrm{dom}(u)| > 0$. We claim that Θ defined from BR as

$$\Theta(u) := \begin{cases} \mathbf{0}_R & \text{if } \neg S_\omega(u) \\ \mathsf{BR}^{\tilde{\phi},\tilde{b},\tilde{\omega}}_{Y,R}(s_u) & \text{otherwise} \end{cases} \tag{12}$$

satisfies the equation (10). The rest of the section contains the lemmas needed to verify this claim. □

The claim that Θ as defined in (12) satisfies equation (10) clearly holds in the case that $\neg S_\omega(u)$, so from now on we assume that $S_\omega(u)$ is true.

The following lemma is not a deep result at all, and can be intuitively seen by inspecting the definition of $s_{u,i}$. However, due to the syntactic complexity of the underlying definitions, its proof is rather long.

Lemma 5.5. *$d(s_{u,i}) = u_i$, for all $i \le |\mathrm{dom}(u)|$.*

Proof. By induction on i. If $i = 0$ the result is trivial, since $d(s_{u,0}) = d(\langle\rangle) = \emptyset$. The induction step is not much more difficult, but involves a deal of tedious verification to perform rigorously. Suppose the lemma holds for some $i < |\mathrm{dom}(u)|$. There are two main cases to deal with: either $n_i < |s_{u,i}|$ or $n_i \ge |s_{u,i}|$.
In the first case we have $s_{u,i+1} = [s_{u,i}](n_i) * \langle u_{i+1}, \mathbf{0}\rangle$. Then for $j < n_i$ we have by definition

$$\begin{aligned} d(s_{u,i+1})(j) &= (\pi_0(s_{u,i+1})_{i'})(j) \text{ for least } i' \le j \text{ such that } j \in \mathrm{dom}(\pi_0(s_{u,i+1})_{i'}), \text{ else undefined} \\ &= (\pi_0(s_{u,i})_{i'})(j) \text{ for least } i' \le j \text{ such that } j \in \mathrm{dom}(\pi_0(s_{u,i})_{i'}), \text{ else undefined} \\ &= d(s_{u,i})(j) \\ &\overset{I.H.}{=} u_i(j) \\ &= u_{i+1}(j) \end{aligned}$$

where we use the fact that $(s_{u,i+1})_{i'} = (s_{u,i})_{i'}$ for $i' \le j < n_i$, and in the last equality that $u_i(j) = u_{i+1}(j)$ fo $j < n_i$. Using similar reasoning, for $j \ge n_i$ we have

$$\begin{aligned} d(s_{u,i+1})(j) &= \begin{cases} (\pi_0(s_{u,i+1})_{i'})(j) \text{ for least } i' < n_i \text{ such that } j \in \mathrm{dom}(\pi_0(s_{u,i+1})_{i'}) \\ (\pi_0(s_{u,i+1})_{n_i})(j) \text{ if } j \in \mathrm{dom}(\pi_0(s_{u,i+1})_{n_i}) \\ \text{else undefined} \end{cases} \\ &\overset{I.H.}{=} \begin{cases} u_i(j) \text{ for least } i' < n_i \text{ such that } j \in \mathrm{dom}(\pi_0(s_{u,i})_{i'}) \\ u_{i+1}(j) \text{ if } j \in \mathrm{dom}(u_{i+1}) \\ \text{else undefined} \end{cases} \\ &= u_{i+1}(j). \end{aligned}$$

For the last equality we use the fact that $j \in \mathrm{dom}(u_i)$ implies $j \in \mathrm{dom}(u_{i+1})$ and $u_i(j) = u_{i+1}(j)$.
For the second main case $n_i \geq |s_{u,i}|$ we have $s_{u,i+1} = s_{u,i} * \langle d(s_{u,i}), f^i_{|s_{u,i}|}\rangle * \ldots * \langle d(s_{u,i}), f^i_{n_i-1}\rangle * \langle u_{i+1}, \mathbf{0}\rangle$, and the argument proceeds as in the first case: for $j < n_i$ we have (expanding the definition of d)

$$\begin{aligned} d(s_{u,i+1}) &= \begin{cases} (\pi_0(s_{u,i})_{i'})(j) \text{ for least } i' \leq j \text{ such that } i' < |s_{u,i}| \text{ and } j \in \mathrm{dom}(\pi_0(s_{u,i})_{i'}) \\ d(s_{u,i})(j) \text{ if } |s_{u,i}| \leq j \text{ and } j \in \mathrm{dom}(d(s_{u,i})) \\ \text{else undefined} \end{cases} \\ &\overset{I.H.}{=} \begin{cases} u_i(j) \text{ for least } i' \leq j \text{ such that } i' < |s_{u,i}| \text{ and } j \in \mathrm{dom}(\pi_0(s_{u,i})_{i'}) \\ u_i(j) \text{ if } |s_{u,i}| \leq j \text{ and } j \in \mathrm{dom}(u_i) \\ \text{else undefined} \end{cases} \\ &= u_i(j) \\ &= u_{i+1}(j). \end{aligned}$$

where for the last equality we use that $u_i(j) = u_{i+1}(j)$ for $j \leq n_i$. For $j = n_i$, it is easy to see that $d(s_{u,i+1})(n_i) = u_{i+1}(n_i)$, since by our assumption $S_\omega(u)$ we know that $n_i \notin \mathrm{dom}(u_i)$ and thus the least $i' \leq n_i$ such that $n_i \in \mathrm{dom}(\pi_0(u_{u,i+1})_{i'})$ is $i' = n_i$. □

Lemma 5.6. *$d(s_u) = u$.*

Proof. Recall that we are assuming $S_\omega(u)$. We have $d(s_u) = d(s_{u,|\mathrm{dom}(u)|}) \overset{\text{L5.5}}{=} u_{|\mathrm{dom}(u)|} \overset{\text{L2.4}}{=} u$. □

Lemma 5.7. *$d^\infty(\widehat{s_u}) = \hat{u}$, and hence $\tilde{\omega}(\widehat{s_u}) = \omega(\hat{u})$.*

Proof. Recall that we assume an encoding of the type $X^\dagger$ such that $\mathbf{0}_{X^\dagger} \equiv \emptyset$, and hence $\mathbf{0}_Y = \langle \mathbf{0}_{X^\dagger}, \mathbf{0}_{X^\dagger \to (X \to R)}\rangle$. It is clear from this that $d^\infty(\hat{s}) = \widehat{d(s)}$ for arbitrary $s \colon Y^*$. Hence, by Lemma 5.6 we have $d^\infty(\widehat{s_u}) = \widehat{d(s_u)} = \hat{u}$ and hence $\tilde{\omega}(\widehat{s_u}) = \omega(d^\infty(\widehat{s_u})) = \omega(\hat{u})$, by the definition of $\tilde{\omega}$. □

Lemma 5.8. *Let $i \leq |\mathrm{dom}(u)|$. Then if $n \notin \mathrm{dom}(u_i)$ and $n < |s_{u,i}|$ then $\pi_1((s_{u,i})_n) = f^k_n$ where k is the least number such that $\forall j \in \{k, \ldots, i-1\}(n_j > n)$.*

Proof. Induction on i. If $i = 0$ then the claim is trivial since $|s_{u,0}| = 0$. Suppose that the lemma is true for $i < |\mathrm{dom}(u)|$. Then as in the proof of Lemma 5.5 there are two main cases corresponding to $n_i < |s_{u,i}|$ or $n_i \geq |s_{u,i}|$.
In the first case, suppose that $n \notin \mathrm{dom}(u_{i+1})$ and $n < |s_{u,i+1}| = n_i + 1$. Then since $n \neq n_i$ (since $n_i \in \mathrm{dom}(u_{i+1})$) we must have $n < n_i$. By $u_i \sqsubset u_{i+1}$ we have $n \notin \mathrm{dom}(u_i)$. Moreover, by our main case assumption that $n_i < |s_{u,i}|$ we can assume that $i > 0$ (else we would have $|s_{u,i}| = 0$), and thus by definition $|s_{u,i}| = n_{i-1} + 1$, and so $n_i \leq n_{i-1}$. By the assumption $S_\omega(u)$ this can be strengthened to $n_i < n_{i-1}$. Therefore we also have $n < |s_{u,i}|$ and by the induction hypothesis obtain that $\pi_1((s_{u,i})_n) = f^k_n$ where k is the least such that $\forall j \in \{k, \ldots, i-1\}(n_j > n)$. Since $n_i > n$, we have $\pi_1((s_{u,i+1})_n) = \pi_1((s_{u,i})_n) = f^k_n$, for the same k, which moreover satisfies the stronger property $\forall j \in \{k, \ldots, i\}(n_j > n)$.
For the second case, suppose again that $n \notin \mathrm{dom}(u_{i+1})$ and $n < |s_{u,i+1}| = n_i$, which as in the first case we can strengthen to $n < n_i$. As in the first case we can also infer that $n \notin \mathrm{dom}(u_i)$. There are two subcases to deal with. Either $n \leq n_{i-1}$ and hence $n \leq |s_{u,i}|$, and so by the induction hypothesis we have $\pi_1((s_{u,i+1})_n) = \pi_1((s_{u,i})_n) = f^k_n$ where k is the least such that $\forall j \in \{k, \ldots, i-1\}(n_j > n)$. Then by the main case assumption we obtain $n_{i-1} + 1 \leq n_i$ and hence $n < n_{i-1} < n_i$, and thus k satisfies the stronger property $\forall j \in \{k, \ldots, i\}(n_j > n)$. Or, in the second subcase $n \geq n_{i-1}$ we have $\pi_1((s_{u,i+1})_n) = f^i_n$. We clearly have $\forall j \in \{k, \ldots, i\}(n_j > n)$ for $i = k$ since this reduces to $n_i > n$ which we have already established. To see that $k = i$ is the least such k, observe that for any $k < i$ we would need $n_{i-1} > n$, which contradicts the premise of the second subcase. □

Lemma 5.9. *If $n \notin \mathrm{dom}(u)$ and $n < |s_u|$ then $\pi_1((s_u)_n) = f^k_n$, where k is the least such that $\forall j \in \{k, \ldots, |\mathrm{dom}(u)| - 1\}(n_j > n)$.*

Proof. This is a direct corollary of Lemma 5.8, setting $i = |\mathrm{dom}(u)|$ and using $u_{|\mathrm{dom}(u)|} = u$ (Lemma 2.4), and recalling that $s_u := s_{u,|\mathrm{dom}(u)|}$. □

Lemma 5.10. *Assuming $\tilde{\omega}(\widehat{s_u}) \geq |s_u|$ we have*

$$\mathsf{BR}^{\tilde{\phi},\tilde{b},\tilde{\omega}}_{Y,R}(s_u) = \begin{cases} b(u) & \text{if } \omega(\hat{u}) \in \mathrm{dom}(u) \\ \phi_u(\lambda x.\mathsf{BR}^{\tilde{\phi},\tilde{b},\tilde{\omega}}_{Y,R}(s_{u\oplus(\omega(\hat{u}),x)})) & \text{otherwise.} \end{cases}$$

Proof. Suppose that $\tilde{\omega}(\widehat{s_u}) \geq |s_u|$ (and $S_\omega(u)$ as in the previous results). Define the sequences $t_{s,m}$, for $m \geq |s_u|$, by course-of-values recursion as

$$t_{s,m} :=_{Y^*} s_u * \langle u, g_{|s_u|}\rangle * \ldots * \langle u, g_{m-1}\rangle$$

where g_k is inductively defined as

$$g_k := \begin{cases} \lambda w, x.\mathsf{BR}(t_{s,k} * \langle [u](k) * x \oplus w, \mathbf{0}\rangle) & \text{if } k \notin \mathrm{dom}(u) \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

As before, let $n = \tilde{\omega}(\widehat{s_u}) = \omega(\hat{u})$. We prove by induction on m that $\mathsf{BR}(s_u) = \mathsf{BR}(t_{s,m})$ for $|s_u| \leq m \leq n+1$. This is true by definition for $m = |s_u|$, so assuming it is also true for arbitrary $m < n+1$ we have

$$\begin{array}{lll}
\mathsf{BR}(s_u) & \overset{\text{(IH)}}{=} & \mathsf{BR}(t_{s,m}) \\
& \overset{(*)}{=} & \tilde{\phi}_{t_{s,m}}(\lambda z\ .\ \mathsf{BR}(t_{s,m} * z)) \\
& \overset{\text{def. } \tilde{\phi}}{=} & \begin{cases} \mathsf{BR}(t_{s,m} * \langle d(t_{s,m}), \mathbf{0}\rangle) & m \in \mathrm{dom}(d(t_{s,m})) \\ \mathsf{BR}(t_{s,m} * \langle d(t_{s,m}), \lambda w, x\ .\ \mathsf{BR}(t_{s,m} * \langle [d(t_{s,m})](m) * x \oplus w, \mathbf{0}\rangle)\rangle) & \text{otherwise} \end{cases} \\
& \overset{(**)}{=} & \begin{cases} \mathsf{BR}(t_{s,m} * \langle u, \mathbf{0}\rangle) & m \in \mathrm{dom}(u) \\ \mathsf{BR}(t_{s,m} * \langle u, \lambda w, x\ .\ \mathsf{BR}(t_{s,m} * \langle [u](m) * x \oplus w, \mathbf{0}\rangle)\rangle) & \text{otherwise} \end{cases} \\
& \overset{\text{def. } g_m}{=} & \mathsf{BR}(t_{s,m} * \langle u, g_m\rangle) \\
& = & \mathsf{BR}(t_{s,m+1})
\end{array}$$

where $(*)$ follows from $\tilde{\omega}(\widehat{t_{s,m}}) = \omega(d^\infty(\widehat{t_{s,m}})) = \omega(\hat{u}) = n \geq m$, with the second equation justified by a simple argument along the lines of Lemma 5.7 (i.e. d^∞ already pick up u in s_u, and cannot acquire any additional elements since the first component of $(\widehat{t_{s,m}})_k$ for $k > |s_u|$ can only be u or $\mathbf{0}$). Equality $(**)$ uses $d(t_{s,m}) = u$, which is proved similarly. By induction we have that $\mathsf{BR}(s_u) = \mathsf{BR}(t_{s,n+1})$. But

$$\tilde{\omega}(\widehat{t_{s,n+1}}) = \omega(d^\infty(\widehat{t_{s,n+1}})) = \omega(\hat{u}) = n < n+1 = |t_{s,n+1}|$$

and therefore

$$\begin{aligned}
\mathsf{BR}(s_u) &= \mathsf{BR}(t_{s,n+1}) \\
&= \tilde{b}(t_{s,n+1}) \\
&= \begin{cases} b(u) & \text{if } n \in \mathrm{dom}(u) \\ \phi_u(\lambda x^X.(\pi_1(t_{s,n+1})_n)(u)(x)) & \text{otherwise.} \end{cases}
\end{aligned}$$

The second equality follows by expanding the definition of $\tilde{b}(t_{s,n+1})$, observing as above that $d(t_{s,n+1}) = u$ and recalling that $n := \omega(\hat{u})$. All that remains to show is that $(\pi_1(t_{s,n+1})_n)(u)(x) = \mathsf{BR}(s_{u\,\oplus\,(n,x)})$ for $n = \omega(\hat{u}) \notin \mathrm{dom}(u)$. This can be shown as

$$\begin{array}{lll}
(\pi_1(t_{s,n+1})_n)(u)(x) & = & g_n(u)(x) \\
& \overset{n\notin\mathrm{dom}(u)}{=} & \mathsf{BR}(t_{s,n} * \langle [u](n) * x \oplus u, \mathbf{0}\rangle) \\
& = & \mathsf{BR}(t_{s,n} * \langle u \oplus (n,x), \mathbf{0}\rangle) \\
& = & \mathsf{BR}(s_{u\,\oplus\,(n,x)})
\end{array}$$

where for the last equality we have (as observed above) $s_{u\,\oplus\,(n,x),|\mathrm{dom}(u)|} = s_u$, and since $n \geq |s_u|$ we have, expanding the definition of $s_{u\,\oplus\,(n,x),|\mathrm{dom}(u)|+1}$ and g_m,

$$s_{u\oplus(n,x)} := s_{u\oplus(n,x),|\mathrm{dom}(u)|+1} = s_u * \langle u, g_{|s_u|}\rangle * \ldots * \langle u, g_{n-1}\rangle * \langle u \oplus (n,x), \mathbf{0}\rangle$$

and therefore $s_{u\,\oplus\,(n,x)} = t_{s,n} * \langle u \oplus (n,x), \mathbf{0}\rangle$. □

Lemma 5.11. *Θ as defined in (12) satisfies equation (10).*

Proof. We only need to consider inputs u that satisfy $S_\omega(u)$, for which we have $\Theta(u) = \mathsf{BR}^{\tilde\phi,\tilde b,\tilde\omega}_{Y,R}(s_u)$. Let us consider the two main cases in the definition of $\mathsf{BR}^{\tilde\phi,\tilde b,\tilde\omega}_{Y,R}$:

(I) If $\tilde\omega(\widehat{s_u}) < |s_u|$ then $\mathsf{BR}^{\tilde\phi,\tilde b,\tilde\omega}_{Y,R}(s_u) = \tilde b(s_u)$. Let $n = \omega(\hat u) = \tilde\omega(\widehat{s_u})$ (the last equality following from Lemma 5.6). Consider two further subcases:

(Ia) If $n \in \mathrm{dom}(u)$ then $\omega(\widehat{d(s_u)}) \in \mathrm{dom}(d(s_u))$ by Lemma 5.6. Hence, by definition of $\tilde b$ we have $\tilde b(s_u) = b(d(s_u)) = b(u)$.

(Ib) If $n \notin \mathrm{dom}(u)$ then $\omega(\widehat{d(s_u)}) \notin \mathrm{dom}(d(s_u))$, again by Lemma 5.6, and we have

$$\begin{aligned}\tilde b(s_u) &= \phi_u(\lambda x.(\pi_1(\widehat{s_u})_n)(u)(x)) \\ &\overset{(*)}{=} \phi_u(\lambda x.\Theta(u \oplus (n,x))),\end{aligned}$$

where for the first equality we use Lemma 5.6 while step $(*)$ is proved as follows. By Lemma 5.9, noting that the premises of the lemma are precisely the assumptions of cases (I) and (Ia), we have

$$(\pi_1(\widehat{s_u})_n)(u)(x) \overset{n<|s_u|}{=} (\pi_1(s_u)_n)(u)(x) = f^k_n(u)(x) \overset{n\notin\mathrm{dom}(u)}{=} \mathsf{BR}([s_{u,k+1}](n) * \langle [v_k](n) * x \oplus u, \mathbf{0}\rangle) \tag{13}$$

where k is the least index such that $\forall j \in \{k, \ldots, |\mathrm{dom}(u)| - 1\}(n_j > n)$. Now, we have

$$[v_k](n) = [d(s_{u,k})](n) \overset{\text{L5.5}}{=} [u_k](n) \overset{(\dagger)}{=} [u](n)$$

where the last equality follows by observing that u is obtained from u_k by updating the latter at points $n_k, \ldots, n_{|\mathrm{dom}(u)|-1}$, but since $n_j > n$ in this range it is clear that $u_k(i) = u(i)$ for $i < n$. Thus $[v_k](n) * x \oplus u = u \oplus (n,x)$. In addition, using again that $n_j > n$ for all $j > k$ and examining definition of $s_{u,i}$ it is easy to see that $[s_{u,k+1}](n) = [s_u](n)$. Therefore

$$\begin{aligned}(\pi_1(\widehat{s_u})_n)(u)(x) &\overset{(13)}{=} \mathsf{BR}([s_u](n) * \langle u \oplus (n,x), \mathbf{0}\rangle) \\ &\overset{(i)}{=} \mathsf{BR}([s_{u\oplus(n,x),|\mathrm{dom}(u)|}](n) * \langle u \oplus (n,x), \mathbf{0}\rangle) \\ &\overset{(ii)}{=} \mathsf{BR}(s_{u\oplus(n,x)}) \\ &= \Theta(u \oplus (n,x))\end{aligned}$$

where (i) follows since the ω-thread of $u \oplus (n,x)$ will coincide with the ω-thread of u up to point n, and thus by extension $s_{u,i}$ will coincide with $s_{u\oplus(n,x),i}$ for $i \le |\mathrm{dom}(u)|$. For (ii) we have

$$\begin{aligned}s_{u\oplus(n,x)} &:= s_{u\oplus(n,x),|\mathrm{dom}(u)|+1} \\ &= [s_{u\oplus(n,x),|\mathrm{dom}(u)|}](n) * \langle u \oplus (n,x), \mathbf{0}\rangle\end{aligned} \tag{14}$$

where in the second equality we expanded the definition of $s_{u\oplus(n,x),|\mathrm{dom}(u)|+1}$, observing that $s_{u\oplus(n,x),|\mathrm{dom}(u)|} = s_{u,|\mathrm{dom}(u)|} = s_u$ and

$$\hat\omega(\{u \oplus (n,x)\}_\omega(|\mathrm{dom}(u)|)) = \hat\omega(u) = n < |s_u| = |s_{u\oplus(n,x),|\mathrm{dom}(u)|}|.$$

For the last equation we used the simple fact that $S_\omega(u) \Rightarrow S_\omega(u \oplus (n,x))$ for $n = \omega(\hat u) \notin \mathrm{dom}(u)$.

(II) Assuming $\tilde\omega(\widehat{s_u}) \ge |s_u|$ we have, by Lemma 5.10, that $\Theta(u) := \mathsf{BR}^{\tilde\phi,\tilde b,\tilde\omega}_{Y,R}(s_u)$ satisfies equation (10), again using that $S_\omega(u) \Rightarrow S_\omega(u \oplus (n,x))$ for $n \notin \mathrm{dom}(u)$, and thus $\Theta(u \oplus (n,x)) = \mathsf{BR}(s_{u\oplus(n,x)})$.
Putting both cases together we have that Θ as defined in (12) satisfies equation (10). □

All that remains to be shown is that the restricted version Θ is equivalent to the full version.

Lemma 5.12. $\mathsf{sBR}_{X,R}$ *is primitive recursively definable from* $\Theta_{X,R}$*, provably in* $\mathsf{E\text{-}HA}^\omega + (\Theta) + \mathsf{sBI}$.

Proof. For some arbitrary input v (not necessarily satisfying $S_\omega(v)$) we define $\mathsf{sBR}^{\phi,b,\omega}(v) := \Theta^{\phi^v,b^v,\omega^v}(\emptyset)$ where

$$\phi_u^v(p^{X\to R}) =_R \begin{cases} b(v\oplus u) & \text{if } \omega(v\oplus\hat{u})\in\mathrm{dom}(v)\\ \phi_{v\oplus u}(p) & \text{otherwise}\end{cases}$$
$$b^v(u) =_R b(v\oplus u)$$
$$\omega^v(\alpha) =_{\mathbb{N}} \omega(v\oplus\alpha).$$

We prove that $\Theta^{\phi^v,b^v,\omega^v}(\emptyset) = \mathsf{sBR}^{\phi,b,\omega}(v)$ by sBI on the predicate

$$P(u) \;\equiv\; \Theta(u) = \mathsf{sBR}(v\oplus u)$$

relative to the functional ω^v. First observe that Θ is sufficient to define the bound $\theta_{\omega,\alpha}(\emptyset)$ of Lemma 2.8, since $\theta_{\omega,\alpha}(\emptyset)$ only makes recursive calls on ω-threads. Therefore provably in $\mathsf{E\text{-}HA}^\omega + (\Theta)$, for all $\alpha\colon X^{\mathbb{N}}$ there exists a least n such that $\widehat{\omega^v}(\{\alpha\}_{\omega^v}(n)) = \hat{\omega}(v\oplus\{\alpha\}_{\omega^v}(n))\in\mathrm{dom}(\{\alpha\}_{\omega^v}(n))$. In this case $\{\alpha\}_{\omega^v}(n)\in S_{\omega^v}$ and therefore

$$\Theta(\{\alpha\}_{\omega^v}(n)) = b^v(\{\alpha\}_{\omega^v}(n)) = b(v\oplus\{\alpha\}_{\omega^v}(n)).$$

But since $\widehat{\omega^v}(\{\alpha\}_{\omega^v}(n))\in\mathrm{dom}(\{\alpha\}_{\omega^v}(n))$ implies $\widehat{\omega^v}(\{\alpha\}_{\omega^v}(n))\in\mathrm{dom}(v\oplus\{\alpha\}_{\omega^v}(n))$ we also have $\mathsf{sBR}(v\oplus\{\alpha\}_{\omega^v}(n)) = b(v\oplus\{\alpha\}_{\omega^v}(n))$. For the induction step, for any $u\in S_{\omega^v}$ we have, expanding Θ and its parameters:

$$\begin{aligned}\Theta(u) &= \begin{cases} b^v(\hat{u}) & \text{if } \omega^v(u)\in\mathrm{dom}(u)\\ \phi_u^v(\lambda x\,.\,\Theta(u\oplus(\omega^v(\hat{u}),x)) & \text{otherwise}\end{cases}\\ &= \begin{cases} b(v\oplus u) & \text{if } \omega(v\oplus\hat{u})\in\mathrm{dom}(u)\\ b(v\oplus u) & \text{if } \omega(v\oplus\hat{u})\in\mathrm{dom}(v)\\ \phi_{v\oplus u}(\lambda x\,.\,\Theta(u\oplus(\omega(v\oplus\hat{u}),x)) & \text{otherwise}\end{cases}\\ &= \begin{cases} b(v\oplus u) & \text{if } \hat{\omega}(v\oplus u)\in\mathrm{dom}(v\oplus u)\\ \phi_{v\oplus u}(\lambda x\,.\,\Theta(u\oplus(\hat{\omega}(v\oplus u),x)) & \text{otherwise}\end{cases}\end{aligned}$$

In the second case, setting $m = \hat{\omega}(v\oplus u) = \omega^v(\hat{u})$, since $u\oplus(m,x)\in S_{\omega^v}$ and $m\notin\mathrm{dom}(u)$ we can assume as induction hypothesis $\forall x P(u\oplus(m,x))$ and thus

$$\begin{aligned}\Theta(u\oplus(m,x)) &= \mathsf{sBR}(v\oplus[u\oplus(m,x)])\\ &= \mathsf{sBR}([v\oplus u]\oplus(m,x))\end{aligned}$$

the first equality following from the bar induction hypothesis, and the last from $m\notin\mathrm{dom}(v)$, and hence $\Theta(u) = \mathsf{sBR}(v\oplus u)$. This establishes the premise of sBI, from which we obtain $P(\emptyset)$, which completes the proof. □

We are now able to prove the main result of the section.

Proof of Theorem 5.2. By Lemma 5.4 a term satisfying the defining equations of $\Theta_{X,R}$ is primitive recursively definable from $\mathsf{BR}_{X^\dagger\times(X^\dagger\to(X\to R)),R}$, therefore by Lemma 5.12, sBR is also primitive recursively definable from $\mathsf{BR}_{X^\dagger\times(X^\dagger\to(X\to R)),R}$, provably in $\mathsf{E\text{-}HA}^\omega + \mathsf{BR} + \mathsf{sBI}$. □

5.3 The BBC functional and further interdefinability results

We conclude this section by putting the interdefinability results in context and taking a closer look at how the symmetric bar recursor compares to functionals from the world of realizability, particularly its symmetric cousin the BBC (Berardi-Bezem-Coquand) functional.

The BBC functional was first defined in [4] in order to give a realizability interpretation to countable choice which has a very natural game-theoretic interpretation (see Section 7 of that paper). Its defining equation is

$$\mathsf{BBC}_{X,R}(u^{X^\dagger}) =_R \phi_u(u\oplus\lambda n.\varepsilon_n(\lambda x.\mathsf{BBC}_{X,R}(u\oplus(n,x)))).$$

This functional has been studied in more detail by Berger [5, 6] and by the second author in [21, 22]. A notable restriction of the BBC functional is that it only gives rise to a total continuous functional when the output type R is restricted to being a base type (whereas $\mathsf{sBR}_{X,R}$ is defined for arbitrary X and R). The proof of totality of BBC given in [5] is of a similar flavour to the domain-theoretic justification of symmetric bar recursion given here in the proof of Theorem 2.11, although there a weak form of Zorn's lemma is required, whereas we use sSpec and our variant of bar induction over ω-threads.

The curious reader is directed to aforementioned works if they wish to learn more about the BBC functional. We mention it here as it is the form of recursion most closely related to our symmetric bar recursor, in that it also takes as an input a partial function $u : X^\dagger$ and makes recursive calls over domain-theoretic extensions of this input. Therefore it is natural to ask how the two modes of recursion compare.

The combined works of several authors [7, 8, 12, 21] has led to the classification up to primitive recursive definability of essentially all forms of recursion used to interpret choice principles. The result is that these recursors fall into two main classes: Those equivalent to Spector's original bar recursion, and those equivalent to the BBC functional (this group includes the more widely used *modified bar recursion*, together with Berger's open and update recursors [6]). The former class is strictly weaker than the latter, in the sense that any functional of the latter class can primitive recursively define any functional of the former, but not vice-versa. The reason for this is that the former group are all S1-S9 definable in the total continuous functionals, while the latter are not. For details of all this, the reader is directed to the papers mentioned above, but for now we simply observe that we are already able to clarify the relationship between symmetric bar recursion and the BBC functional from the perspective of interdefinability: the main results of this section prove that symmetric bar recursion belongs to the weaker class of recursors as it is equivalent to Spector's bar recursion, and is therefore strictly weaker than the BBC functional.

There is also a key qualitative difference between the algorithms which underlie the two forms of recursion. The the defining equation of the BBC functional, both of the variables n and x fall under the scope of a λ-abstraction. What this means is that when we want to compute $\mathsf{BBC}(u)$, it is possible that we will make recursive calls on extensions $u \oplus (n, x)$ of u for a range of values of $n \notin \mathrm{dom}(u)$. On the other hand, symmetric bar recursion only makes recursive calls on extensions of the form $u \oplus (\hat{\omega}(u), x)$ i.e. at a specific point determined by the functional ω. Thus the recursion carried out by sBR takes place along a tree whose branches are ω-threads (as defined in Section 2), whereas the recursion carried out by BBC takes place over a tree whose branches can consist of *arbitrary* sequences of updates. Therefore despite the obvious cosmetic similarity between sBR and BBC, symmetric bar recursion is in this sense more far closely related to normal bar recursion, which also is restricted to updates at a particular point, in this case only extensions $s * x$ of finite sequences s.

The efficiency of BBC relative to the other forms of recursion in its class, such as modified bar recursion or the more recent *implicit product of selection functions* [12], is still very much unknown, and it could well be the case that the additional λ-abstraction over the variable n could in fact give rise to realisers vastly more complex to those built from the sequential forms of recursion. However, we leave such questions to future research.

6 From Countable Choice to Discrete Choice

In this final section we show how our move from BR to sBR opens the door to extending bar recursion to more complex types, thus leading to a computational interpretation of a wider class of choice principles. Up until now, we have considered bar recursion over either finite sequences or finite partial functions: in other words over objects of type $\mathbb{N} \to X + \mathbf{1}$ with finite support. It is natural ask whether we can further generalise bar recursion to take as input objects of type $D \to X + \mathbf{1}$ with finite support, for some suitable class of indexing types D.

It is clear that for such a bar recursor to be well-defined we require equality on D to be decidable, and moreover for well-foundedness we require that the stopping condition $\omega(\hat{u}) \in \mathrm{dom}(u)$ is eventually reached for u with sufficiently large domain. This first condition is already highly restrictive: in PA^ω decidability of equality is only guaranteed for types of lowest level. However, it has been shown by Escardó [11] that in the Kleene-Kreisel continuous functionals, decidability of equality can be established for a wider class of types.

Definition 6.1 (Escardó [11]). Inductively define the *discrete* and *compact* types by

$$\begin{aligned}\mathsf{discrete} &::= \mathbb{B} \mid \mathbb{N} \mid \mathsf{discrete} \times \mathsf{discrete} \mid \mathsf{compact} \to \mathsf{discrete}\\ \mathsf{compact} &::= \mathbb{B} \mid \mathsf{compact} \times \mathsf{compact} \mid \mathsf{discrete} \to \mathsf{compact}.\end{aligned}$$

This terminology is based on the fact that the space C_X of Kleene-Kreisel continuous functionals of type X is discrete under the usual (quotient of the) Scott topology whenever X is a discrete type, and is compact whenever X is a compact type. Several properties of discrete and compact types are established in [11], including the fact that for *arbitrary* discrete X the space C_X is both computably enumerable and has decidable equality[6] (this is striking given that the discrete types contain genuine higher-types such as $\mathbb{B}^{\mathbb{N}} \to \mathbb{N}$). Moreover, by a standard topological argument one can extend the usual sequential continuity property for functionals on infinite sequences to the following principle:

$$\mathsf{Cont}[D] \;:\; \forall \omega^{X^D \to D}, \alpha^{X^D} \exists S \subset D \forall \beta (\forall d \in S(\alpha(d) =_X \beta(d)) \to \omega(\alpha) =_D \omega(\beta)),$$

where S is a finite subset of the discrete type D.

From now on let D range over discrete types. Let $X^{\dagger D}$ denote the type of finite partial functions from D to X, i.e. partial functions $u\colon D \rightharpoonup X$ with finite domain. We define $\mathsf{sBR}[D]$ where D ranges over discrete types by

$$\mathsf{sBR}[D]^{\phi,b,\omega}_{X,R}(u^{X^{\dagger D}}) =_R \begin{cases} b(u) & \text{if } \hat{\omega}(u) \in \mathrm{dom}(u) \\ \phi_u(\lambda x \,.\, \mathsf{sBR}[D]^{\phi,b,\omega}(u \oplus (\omega(\hat{u}), x))) & \text{otherwise} \end{cases}$$

where $u\colon X^{\dagger D}$ and $\omega\colon (D \to X) \to D$. Note that sBR as defined in Section 2 is just $\mathsf{sBR}[\mathbb{N}]$. This generalised form of bar recursion is well-defined in the continuous functionals since equality on D is decidable, and therefore the constructions $(\hat{\cdot})\colon X^{\dagger D} \to X^D$ and $\oplus : X^{\dagger D} \to (D \times X) \to X^{\dagger D}$ are still continuous (which would not be the case for e.g. $D = \mathbb{N} \to \mathbb{N}$).

Moreover, the recursor $\mathsf{sBR}[D]$ is well founded by $\mathsf{Cont}[D]$: Suppose that $\mathsf{sBR}(u)$ is not total for some total input u. Then by classical dependent choice we can construct a sequence recursively by $u_0 := u$ and $u_{i+1} := u_i \oplus (d_i, x_i)$, where for each i we have

$$(i)\;\; d_i = \hat{\omega}(u_i) \notin \mathrm{dom}(u_i) \qquad (ii)\;\; \mathsf{sBR}(u_i) \text{ is not total.}$$

By classical countable choice define $\alpha\colon D \to X + \mathbf{1}$ by $\alpha(d) := x_i$ if $d = d_i$ for some i, and undefined otherwise. Then by $\mathsf{Cont}[D]$ there exists a finite subset $S \subset D$ such that $\omega(\alpha) = \omega(\beta)$ whenever $\alpha(d) = \beta(d)$ for all $d \in S$. Now since α is the domain-theoretic limit of the u_i, there is some index I such that $u_I(d) = \alpha(d)$ for $d \in S$, and therefore $d_I = \hat{\omega}(u_I) = \omega(\alpha)$. Now by definition we have $d_I \in \mathrm{dom}(u_{I+1})$, and so in particular $d_I \in \mathrm{dom}(\alpha)$. It is clear that $d_I \notin S$, since by (i) we have $d_I \notin \mathrm{dom}(u_I)$ but $d_I \in \mathrm{dom}(\alpha)$, contradicting the definition of I. But then $u_{I+1} = u_I$ on S, and therefore $d_{I+1} = \omega(\alpha) = d_I$ and hence $d_{I+1} \in \mathrm{dom}(u_{I+1})$, a contradiction.

This constitutes an informal argument that $\mathsf{sBR}[D]$ is a well-defined, total continuous functional, and so by an entirely analogous procedure to the case of $\mathsf{sBR}[\mathbb{N}]$, one can construct f and p in $\mathsf{sBR}[D]$ which satisfy the appropriate generalisation of Spector's equations:

$$\begin{aligned}\omega f &=_D d\\ f(\omega f) &=_X \varepsilon_d p\\ qf &=_Y p(\varepsilon_d p).\end{aligned}$$

As a result, we gain a computational interpretation of the following axiom of *discrete choice*:

$$\mathsf{AC}_{D,X} \;:\; \forall d^D \exists x^X A(d,x) \to \exists f^{D \to X} \forall d A(d, fd).$$

We remark that, as shown in [11], the set C_D is recursively enumerable for any discrete type D, and so can be encoded in the usual type of natural numbers $\mathbb{N}$. Therefore in theory we could have defined both $\mathsf{sBR}[D]$ and the analogous generalisation $\mathsf{BR}[D]$ of Spector's bar recursion (where the points of D

[6] However, equality may not be *primitive recursively* decidable as in PA^ω: for non-trivial discrete types one must appeal to the so-called *infinite product of selection functions* (see [11] for details).

are ordered relative to their encoding in $\mathbb{N}$) in terms of $\mathsf{sBR}[\mathbb{N}]$ and $\mathsf{BR}[\mathbb{N}]$ respectively. However, this reduction to the base level would of course rely explicitly on the encoding of C_D into $\mathbb{N}$ on the meta-level, which is not guaranteed to be primitive recursive. To avoid this extra layer of complexity, and to define the generalised recursor directly seems only possible for the symmetric recursor $\mathsf{sBR}[D]$, as Spector's bar recursion relies inherently on the underlying ordering of the natural numbers, and is therefore prime-facie undefined for higher level discrete types on which no natural total ordering exists.

Note that whereas in the main part of the paper, the proposed benefits of our symmetric recursor lay in its intensional aspects – notably the efficiency of its underlying algorithm – here it is its purely extensional properties, namely that its defining equation does not assume any ordering on D, that are found to be useful.

7 Conclusion

We have introduced a variant of bar recursion that, unlike Spector's bar recursion, carries out recursive calls in symmetric manner, as dictated by the control parameter. We have shown that this symmetric recursor exists in the usual models of bar recursion, such as the Kleene-Kreisel continuous functionals, and is in fact primitive recursively equivalent to Spector's bar recursion. We then showed that Spector's equations, a solution to which is sufficient to give a Dialectica interpretation to the negated axiom of countable choice, can be solved with a special case of symmetric bar recursion, analogously to Spector's original bar recursive solution. We compared concrete realizers obtained from the classical proof that there is no injection from $\mathbb{N}^{\mathbb{N}} \to \mathbb{N}$ using both variants of bar recursion, and demonstrated that our new method of extracting programs from proofs in classical analysis performs dramatically more efficiently in many cases.

Our work fits in to the much broader program of adapting and refining traditional proof theoretic techniques so that they are better suited to their role in modern proof theory - in our case taking a well-known method of proving the consistency of classical analysis and altering it so that it becomes better suited as a tool for extracting programs from proofs. However, concrete applications of program extraction using symmetric bar recursion are currently restricted to the single case study given here, and we believe that these others more attention in the future.

In addition to the extension of bar recursion to discrete choice principles sketched above, it would be particularly interesting to investigate the procedural behaviour of symmetric bar recursion, which seems much more natural than Spector's bar recursion and has close links to both the update procedures of Avigad [2] and the learning-based realizers of Aschieri et al. [1]. Making this relationship precise could lead to a much better understanding of the relationship between proof interpretations like the Dialectica interpretation and more direct learning-based interpretations of classical logic.

Acknowledgements. The authors would like to thank to anonymous referee, whose many comments and corrections led to a much improved version of the paper. This work was carried out while the second author was supported by the Austrian Science Fund (FWF) project P 25781-N15.

References

[1] F. Aschieri and S. Berardi. Interactive learning-based realizability for Heyting arithmetic with $\mathsf{EM1}$. *Logical Methods in Computer Science*, 6(3), 2010.

[2] J. Avigad. Update procedures and the 1-consistency of arithmetic. *Mathematical Logic Quarterly*, 48(1):3–13, 2002.

[3] 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.

[4] 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.

[5] U. Berger. The Berardi-Bezem-Coquand functional in a domain-theoretic setting. Unpublished results, available from author's webpage at http://www.cs.swan.ac.uk/∼csulrich/recent-papers.html, 2002.

[6] U. Berger. A computational interpretation of open induction. In F. Titsworth, editor, *Proceedings of the Nineteenth Annual IEEE Symposium on Logic in Computer Science*, pages 326–334. IEEE Computer Society, 2004.

[7] U. Berger and P. Oliva. Modified bar recursion and classical dependent choice. *Lecture Notes in Logic*, 20:89–107, 2005.

[8] U. Berger and P. Oliva. Modified bar recursion. *Mathematical Structures in Computer Science*, 16(2):163–183, 2006.

[9] M. Bezem. Strongly majorizable functionals of finite type: A model for bar recursion containing discontinuous functionals. *Journal of Symbolic Logic*, 50:652–660, 1985.

[10] Y. L. Ershov. Model C of partial continuous functionals. In *Logic Colloquium*, pages 455–467. North Holland, Amsterdam, 1977.

[11] M. Escardó. Exhaustible sets in higher-type computation. *Logical Methods in Computer Science*, 4(3):1–37, 2008.

[12] M. Escardó and P. Oliva. Bar recursion and products of selection functions. *Journal of Symbolic Logic*, 80(1):1–28, 2015.

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

[14] W. A. Howard. Functional interpretation of bar induction by bar recursion. *Compositio Mathematica*, 20:107–124, 1968.

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

[16] U. Kohlenbach. *Applied Proof Theory: Proof Interpretations and their Use in Mathematics*. Monographs in Mathematics. Springer, 2008.

[17] G. Kreisel. Interpretation of analysis by means of functionals of finite type. In A. Heyting, editor, *Constructivity in Mathematics*, pages 101–128. North-Holland, Amsterdam, 1959.

[18] P. Oliva. Understanding and using Spector's bar recursive interpretation of classical analysis. In A. Beckmann, U. Berger, B. Löwe, and J. V. Tucker, editors, *Proceedings of CiE'2006*, volume 3988 of *LNCS*, pages 423–234, 2006.

[19] P. Oliva and T. Powell. On Spector's bar recursion. *Mathematical Logic Quarterly*, 58:356–365, 2012.

[20] T. Powell. *On Bar Recursive Interpretations of Analysis*. PhD thesis, Queen Mary University of London, 2013.

[21] T. Powell. The equivalence of bar recursion and open recursion. *Annals of Pure and Applied Logic*, (165):1727–1754, 2014.

[22] T. Powell. Parametrised bar recursion: A unifying framework for realizability interpretations of classical dependent choice. Journal of Logic and Computation, 2015. Advance access at http://logcom.oxfordjournals.org/content/early/2015/08/28/logcom.exv056.abstract.

[23] B. Scarpellini. A model for bar recursion of higher types. *Compositio Mathematica*, 23:132–153, 1971.

[24] 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.

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