THE JOURNAL OF SYMBOLIC LOGIC
Volume 82, Number 2, June 2017

THE HERBRAND FUNCTIONAL INTERPRETATION OF THE DOUBLE NEGATION SHIFT

MARTÍN ESCARDÓ AND PAULO OLIVA

Abstract. This paper considers a generalisation of selection functions over an arbitrary strong monad T, as functionals of type $J_R^T X = (X \to R) \to TX$. It is assumed throughout that R is a T-algebra. We show that J_R^T is also a strong monad, and that it embeds into the continuation monad $K_R X = (X \to R) \to R$. We use this to derive that the explicitly controlled product of T-selection functions is definable from the explicitly controlled product of quantifiers, and hence from Spector's bar recursion. We then prove several properties of this product in the special case when T is the finite powerset monad $\mathcal{P}_f(\cdot)$. These are used to show that when $TX = \mathcal{P}_f(X)$ the explicitly controlled product of T-selection functions calculates a witness to the Herbrand functional interpretation of the double negation shift.

§1. Introduction.

Gödel's *functional* or *Dialectica* interpretation was introduced in [10] as a reduction of first order arithmetic to the "finitistic" quantifier-free calculus of primitive recursive functionals (system T). Soon after Gödel's paper appeared in print, Spector [17] showed how Gödel's interpretation of arithmetic could be extended to analysis by extending system T with what he called *bar recursion*. By *analysis* we mean classical arithmetic in all finite types extended with countable choice and dependent choice—and hence comprehension.

Spector's original work has given rise to several other bar recursive interpretations of analysis, whereby different proof interpretations other than the Dialetica interpretation have been used. In such cases one was either able to continue using Spector's original form of bar recursion (e.g., [8,12]) or some variant of bar recursion was proposed (e.g., [1, 2]).

The notion of a *selection function* as used in this paper was first introduced in [5], where it was shown to be related to bar recursion and the calculation of optimal strategies in a general class of sequential games. The connections with game theory and bar recursion were further developed in [6] and [7], respectively. These papers, and the present one, also make use of the notion of *quantifiers*—a further generalisation on Mostowski generalised quantifies [14]. The main result of [7] is that Spector's original bar recursion can be shown to be equivalent to the iterated product of quantifiers, whereas the restricted form needed to witness the Dialectica interpretation of DNS is equivalent to the iterated product of selection functions.

The correspondence between computability and games is based on the modelling of players via quantifiers $K_R X = (X \to R) \to R$. If X is the set of moves available

Received October 19, 2015.
2010 *Mathematics Subject Classification.* 03F07, 03F10.
Key words and phrases. bar recursion, Herbrand functional interpretation, dialectica interpretation, monads.

to a player, and R is the set of possible outcomes, then mappings of type $X \to R$ can be seen as describing the context a player lives in. Such contexts (a form of continuation) describe the final outcome for each of the possible choices of the player. Hence, to specify a player is to describe her preferred outcomes for each given game context. Similarly, a selection function $J_R X = (X \to R) \to X$ also takes a game context as input, but determines the optimal move for any given game context.

In this paper we consider the iterated product of selection functions parametrised by an arbitrary strong monad TX, i.e., $J_R^T X = (X \to R) \to TX$. Using the intuition that an element of a monad TX provides "information" about concrete elements of X, and the correspondence with games, we can view such selection functions $J_R^T X$ as specifying some information about the optimal move for any given game context.

We study the bar recursion that arises from the iterated product of such T-selection functions. Our first step is to show that $J_R^T X$ is also a strong monad. Since any strong monad embeds into the continuation monad, it follows that we have an embedding of $J_R^T X$ into KX. We make use of this embedding to show that the iterated product of T-selection functions is in fact primitive recursively definable from the iterated product of quantifiers, and hence from Spector's original bar recursion.

Finally, we consider the particular case when TX is the finite powerset monad $\mathcal{P}_{\mathrm{f}}(X)$. We prove several properties of the iterated product of selection functions $(X \to R) \to \mathcal{P}_{\mathrm{f}}(X)$, and show how it provides a witness for the *Herbrand functional interpretation* [19] of double-negation shift DNS

$$\forall n^{\mathbb{N}} \neg\neg A(n) \to \neg\neg\forall n^{\mathbb{N}} A(n).$$

1.1. Heyting arithmetic in all finite types, and bar induction. We work in the setting of Heyting arithmetic in all finite types, with full extensionality. This corresponds to the system E-HA$^{\omega}$ of [18]. When carrying out the verification of the Herbrand functional interpretation of DNS we will make free use of classical logic, in order to simplify the verification of the bar-recursive construction, hence will be working on E-PA$^{\omega}$. Although it is well-known that full extensionality is not normally interpreted by the functional interpretations, we are simply assuming full extensionality in the verification of our interpretation of DNS, which is obviously harmless.

The quantifier-free part of the theories E-HA$^{\omega}$ and E-PA$^{\omega}$ is normally referred to as Gödel's system T. Although in T one normally only assumes the natural numbers $\mathbb{N}$ as basic types, and function space constructions $X \to Y$ as the only type constructor, we will follow here the same formulation of T as in [19] where one also assumes products $X \times Y$, finite sequences X^*, and even finite powersets $\mathcal{P}_{\mathrm{f}}(X)$. We write $r \preceq s$ to say that the finite sequence r is a prefix of the finite sequence s. We assume that each type X contains a 'default' value $\mathbf{0} \colon X$, so that we can define an canonical extension operation $(\cdot)^+ \colon X^* \to X^{\mathbb{N}}$ from finite to infinite sequences, by appending an infinite sequence of default values. For instance, for the natural numbers $\mathbf{0}^{\mathbb{N}}$ could be the number zero, whereas for $\mathcal{P}_{\mathrm{f}}(X)$ we can take $\mathbf{0}^{\mathcal{P}_{\mathrm{f}}(X)} = \emptyset$.

On top of E-HA$^{\omega}$, in the proofs of Lemmas 3.2 and 3.3 will make use of the following form of bar induction:

Definition 1.1 (Bar induction). Let $P(s)$ be a universal formula. We say that bar induction holds for $P(s)$ if for any functional $\omega\colon X^{\mathbb{N}} \to \mathbb{N}$, $P(\langle\rangle)$ holds whenever the following two conditions hold for all $s\colon X^*$

- $\omega(s^+) < |s|$ implies $P(s)$, and
- $\omega(s^+) \geq |s|$ and $\forall x P(s * x)$ implies $P(s)$.

This form of bar induction implicitly assumes that the bar condition $\omega(s^+) < |s|$ eventually holds. This is indeed the case in all models of Spector's bar recursion [3, 16].

Notation 1.2. *In the paper we will use sub-scripts in four different ways, and hope their respective meanings will be clear from context*:

- *In the following section we use sub-scripts to denote the type of a functional. For instance, the identity function of type X will be written as* id_X.
- *If $\alpha\colon X \to Y$ we can view α as a family of elements of Y indexed by X, i.e., $\{\alpha_x\}_{x\colon X}$. When taking this view we might write α_x instead of $\alpha(x)$.*
- *Bar recursive functionals have several parameters, normally* $\mathsf{BR}(\omega)(s)(\varepsilon)(q)$. *In order to focus on the selection functions ε and the outcome function q we shall rewrite this as* $\mathsf{BR}_s^{\omega}(\varepsilon)(q)$. *This makes sense since s is the 'index' of the bar recursion whereas ω is the stopping condition.*
- *Finally, for $q\colon X^* \to R$ we write $q_s(t)$ for $q(s * t)$ when we wish to 'partially evaluate' q on s to produce another function $q_s\colon X^* \to R$.*

1.2. Strong monads. In this section we recall the basic notions about strong monads needed in this paper. Throughout the paper we work in Gödel's system T. Hence, X, Y, and R should be viewed as finite types.[1]

Definition 1.3 (Strong monad). Let T be a meta-level unary operation on types, that we will call a *type operator*. A type operator T is called a *strong monad* if we have a family of closed terms

$$\begin{aligned} \eta_X &: X \to TX \\ (\cdot)^\dagger &: (X \to TY) \to (TX \to TY) \end{aligned}$$

satisfying (provably in T) the laws

(i) $(\eta_X)^\dagger = \mathrm{id}_{TX}$,

(ii) $g^\dagger \circ \eta_Y = g$,

(iii) $(g^\dagger \circ f)^\dagger = g^\dagger \circ f^\dagger$,

where $g\colon Y \to TR$ and $f\colon X \to TY$. When several strong monads are involved we shall use the super-script in η^T so as to be clear which η is being used.

Given $f\colon X \to Y$ we define $Tf\colon TX \to TY$ by $Tf = (\eta_Y \circ f)^\dagger$. The laws for the monad show that this construction makes T into a functor, that is, $T\,\mathrm{id}_X = \mathrm{id}_{TX}$, and for $g : Y \to Z$ we have $T(g \circ f) = Tg \circ Tf$.

Monads have been extensively studied in category theory [11], programming language semantics [13], and in the functional programming community [20].

[1] It will be clear, however, that what we describe would work more generally in any of the well-known models of higher-order computability.

In a monad one would normally have a nonuniform mapping from $f\colon X \to TY$ to $f^\dagger\colon TX \to TY$. The term *strong* here refers to the assumption that we have a uniform map $(\cdot)^\dagger\colon (X \to TY) \to (TX \to TY)$.

Example 1.4 (Selection functions and quantifiers). The simplest strong monad is the identity $TX = X$, where $\eta_X(x) = x$ and $f^\dagger = f$. As shown in [5], for any fixed type R, the type operators $K_R X = (X \to R) \to R$, called *quantifiers*, and $J_R X = (X \to R) \to X$, called *selection functions*, are strong monads.

Example 1.5 (Finite powerset monad). Assuming a definitional extension of Gödel's system T with the finite nonempty powerset type constructor $\mathcal{P}_{\mathrm{f}}(X)$, and associated operations, we also have the finite nonempty powerset monad $TX = \mathcal{P}_{\mathrm{f}}(X)$, with

- $\eta_X(x) = \{x\}$,
- $f^\dagger(S) = \bigcup_{x\in S} f(x)$, for $f\colon X \to \mathcal{P}_{\mathrm{f}}(Y)$.

This particular strong monad will be further studied in Section 4. Note that if we include the empty set, or even consider the full powerset, the resulting structure is also a monad. For the purposes of our application in Section 4 it is irrelevant whether the empty set is included or not, but we do need the restriction to finite sets.

Definition 1.6 (T-algebra). Given a strong monad T, a type R is called a T-*algebra* if we have a family of maps $(\cdot)^*\colon (X \to R) \to (TX \to R)$ satisfying

(i) $g^* \circ \eta_Y = g$,

(ii) $(g^* \circ f)^* = g^* \circ f^\dagger$,

where $g\colon Y \to R$ and $f\colon X \to TY$.

It is a trivial fact that for any monad T, the type TY is always a T-algebra, with the map $(\cdot)^* = (\cdot)^\dagger\colon (X \to TY) \to (TX \to TY)$. That this map satisfies the T-algebra equations (i) and (ii) also follows directly from the assumption that it satisfies the monad equations (ii) and (iii).

The reason we focus here on *strong* monads is that on such monads we can define a binary product operation as follows:

Definition 1.7 (Monadic product). For any strong monad T define its *product operation* $\otimes^T \colon TX \times (X \to TY) \to T(X \times Y)$ as

$$a \otimes^T f = (\lambda x.(\lambda y.\eta_{X\times Y}(x, y))^\dagger(fx))^\dagger(a). \tag{1}$$

When the monad in question is clear from the context we will omit the subscript in $\otimes^T$.

Example 1.8. Given η_X and $(\cdot)^\dagger$ as described in Example 1.5, the monadic product for the finite powerset monad can be shown to be defined as

$$a \otimes f = \bigcup_{x\in a} \bigcup_{y\in f(x)} \{\langle x, y\rangle\}.$$

Lemma 1.9. *The monadic product operation satisfies, for $q\colon X \times Y \to TR$, $a\colon TX$ and $f\colon X \to TY$*

$$q^\dagger(a \otimes f) = (\lambda x.(q_x)^\dagger(fx))^\dagger(a),$$

where $q_x = \lambda y.q(x, y)$. When $q \colon X \times Y \to R$ and R is a T-algebra it also satisfies

$$q^*(a \otimes f) = (\lambda x.(q_x)^*(fx))^*(a).$$

Proof. We calculate as follows:

$$\begin{aligned}
q^\dagger(a \otimes f) &\overset{(1)}{=} q^\dagger((\lambda x.(\lambda y.\eta_{X\times Y}(x, y))^\dagger(fx))^\dagger(a))\\
&\overset{(\circ)}{=} (q^\dagger \circ (\lambda x.(\lambda y.\eta_{X\times Y}(x, y))^\dagger(fx))^\dagger)(a)\\
&\overset{\text{D1.3}(iii)}{=} (q^\dagger \circ (\lambda x.(\lambda y.\eta_{X\times Y}(x, y))^\dagger(fx)))^\dagger(a)\\
&\overset{(\circ)}{=} (\lambda x.q^\dagger((\lambda y.\eta_{X\times Y}(x, y))^\dagger(fx)))^\dagger(a)\\
&\overset{(\circ)}{=} (\lambda x.((q^\dagger \circ (\lambda y.\eta_{X\times Y}(x, y))^\dagger)(fx)))^\dagger(a)\\
&\overset{\text{D1.3}(iii)}{=} (\lambda x.((q^\dagger \circ (\lambda y.\eta_{X\times Y}(x, y)))^\dagger(fx)))^\dagger(a)\\
&\overset{(\circ)}{=} (\lambda x.((\lambda y.q^\dagger(\eta_{X\times Y}(x, y)))^\dagger(fx)))^\dagger(a)\\
&\overset{\text{D1.3}(ii)}{=} (\lambda x.(q_x)^\dagger(fx))^\dagger(a).
\end{aligned}$$

In the case $q \colon X \times Y \to R$ and R is a T-algebra we use properties (i) and (ii) of Definition 1.6 instead. ⊣

§2. T-Selection functions.

In the following two sections we assume that T is a strong monad, and that R is a T-algebra.

Definition 2.1 (T-selection functions). Let $J_R^T X = (X \to R) \to TX$, where R is a T-algebra. The elements of the type $J_R^T X$ will be called T-*selection functions*.

Under the assumptions that T is a strong monad and R a T-algebra, it follows that J_R^T is also a strong monad.

Lemma 2.2. *J_R^T is a strong monad with operations*:

(i) $\eta_X^{J_R^T}(x) = \lambda p.\eta_X^T(x)$,

(ii) $\delta^\dagger(\varepsilon) = \lambda p.(b_p^\delta)^\dagger(a_p^{\varepsilon,\delta})$, *where* $\delta \colon X \to J_R^T Y$ *and* $\delta^\dagger \colon J_R^T X \to J_R^T Y$,

where $b_p^\delta(x) \overset{TY}{=} \delta(x)(p)$ *and* $a_p^{\varepsilon,\delta} \overset{TX}{=} \varepsilon(p^* \circ b_p^\delta)$.

Proof. We must check condition (i)–(iii) of Definition 1.3. Let us start with condition (i), defining $\delta(x) :\equiv \lambda p.\eta_X^T(x)$,

$$\begin{aligned}
(\eta_X^{J_R^T})^\dagger(\varepsilon)(p) &\overset{\eta_X^{J_R^T}\ \text{def.}}{=} (\lambda x\lambda p.\eta_X^T(x))^\dagger(\varepsilon)(p)\\
&\overset{(\cdot)^\dagger\ \text{def.}}{=} (b_p^\delta)^\dagger(a_p^{\varepsilon,\delta})\\
&\overset{b\ \text{def.}}{=} (\eta_X^T)^\dagger(a_p^{\varepsilon,\delta})\\
&\overset{\text{D1.3(i)}}{=} a_p^{\varepsilon,\delta}\\
&\overset{a\ \text{def.}}{=} \varepsilon(p^* \circ b_p^\delta)\\
&\overset{b,\delta\ \text{def.}}{=} \varepsilon(p^* \circ \eta_X^T)\\
&\overset{\text{D1.3(ii)}}{=} \varepsilon(p).
\end{aligned}$$

For condition (ii) we have

$$\begin{aligned}(\delta^\dagger \circ \eta_Y^{J_R^T})(y)(p) &= \delta^\dagger(\lambda p.\eta_Y^T(y))(p)\\ &\overset{(\cdot)^\dagger \text{ def.}}{=} (b_p^\delta)^\dagger(a_p^{\lambda p.\eta_Y^T(y),\delta})\\ &\overset{a \text{ def.}}{=} (b_p^\delta)^\dagger(\eta_Y^T(y)) \overset{\text{D1.3(ii)}}{=} b_p^\delta(y) \overset{b \text{ def.}}{=} \delta(y)(p).\end{aligned}$$

Finally, define $\Delta_x(p) = (b_p^\delta)^\dagger(a_p^{\varepsilon_x,\delta})$ and $\Gamma_v(p) = (b_p^\varepsilon)^\dagger(a_p^{v,\varepsilon})$. We have:

$$\begin{aligned}(\delta^\dagger \circ \varepsilon)^\dagger(v) &= (\lambda x.\delta^\dagger(\varepsilon_x))^\dagger(v)\\ &\overset{(\cdot)^\dagger \text{ def.}}{=} (\lambda x\lambda p.(b_p^\delta)^\dagger(a_p^{\varepsilon_x,\delta}))^\dagger(v)\\ &\overset{\Delta \text{ def.}}{=} (\lambda x.\Delta_x)^\dagger(v)\\ &\overset{(\cdot)^\dagger \text{ def.}}{=} \lambda q.(b_q^\Delta)^\dagger(a_q^{v,\Delta})\\ &\overset{b \text{ def.}}{=} \lambda q.(\lambda x.\Delta_x(q))^\dagger(a_q^{v,\Delta})\\ &\overset{\Delta \text{ def.}}{=} \lambda q.((b_q^\delta)^\dagger \circ (\lambda x.a_q^{\varepsilon_x,\delta}))^\dagger(a_q^{v,\Delta})\\ &\overset{\text{D1.3(iii)}}{=} \lambda q.((b_q^\delta)^\dagger \circ (\lambda x.a_q^{\varepsilon_x,\delta})^\dagger)(a_q^{v,\Delta})\\ &= \lambda q.(b_q^\delta)^\dagger((\lambda x.a_q^{\varepsilon_x,\delta})^\dagger(a_q^{v,\Delta}))\\ &\overset{(*)}{=} \lambda q.(b_q^\delta)^\dagger(a_q^{\Gamma_v,\delta})\\ &\overset{(\cdot)^\dagger \text{ def.}}{=} \delta^\dagger(\Gamma_v)\\ &\overset{\Gamma \text{ def.}}{=} \delta^\dagger(\lambda p.(b_p^\varepsilon)^\dagger(a_p^{v,\varepsilon})) \overset{(\cdot)^\dagger \text{ def.}}{=} \delta^\dagger(\varepsilon^\dagger(v)) = (\delta^\dagger \circ \varepsilon^\dagger)(v).\end{aligned}$$

It remains to show that $(*)$ $a_q^{\Gamma_v,\delta} = (\lambda x.a_q^{\varepsilon_x,\delta})^\dagger(a_q^{v,\Delta})$. This can be shown as

$$\begin{aligned}a_q^{\Gamma_v,\delta} &\overset{a \text{ def.}}{=} \Gamma_v(q^* \circ b_q^\delta)\\ &\overset{\Gamma \text{ def.}}{=} (b_{q^*\circ b_q^\delta}^\varepsilon)^\dagger(a_{q^*\circ b_q^\delta}^{v,\varepsilon})\\ &\overset{b \text{ def.}}{=} (\lambda x.\varepsilon_x(q^* \circ b_q^\delta))^\dagger(a_{q^*\circ b_q^\delta}^{v,\varepsilon})\\ &\overset{(**)}{=} (\lambda x.\varepsilon_x(q^* \circ b_q^\delta))^\dagger(a_q^{v,\Delta})\\ &\overset{a \text{ def.}}{=} (\lambda x.a_q^{\varepsilon_x,\delta})^\dagger(a_q^{v,\Delta}),\end{aligned}$$

where, finally, $(**)$ $a_{q^*\circ b_q^\delta}^{v,\varepsilon} = a_q^{v,\Delta}$ is shown as

$$\begin{aligned}a_{q^*\circ b_q^\delta}^{v,\varepsilon} &\overset{a \text{ def.}}{=} v((q^* \circ b_q^\delta)^* \circ b_{q^*\circ b_q^\delta}^\varepsilon)\\ &\overset{\text{D1.6(ii)}}{=} v(q^* \circ (b_q^\delta)^\dagger \circ b_{q^*\circ b_q^\delta}^\varepsilon)\\ &= v(\lambda x.q^*((b_q^\delta)^\dagger(b_{q^*\circ b_q^\delta}^\varepsilon(x))))\\ &\overset{b \text{ def.}}{=} v(\lambda x.q^*((b_q^\delta)^\dagger(\varepsilon_x(q^* \circ b_q^\delta))))\\ &\overset{a \text{ def.}}{=} v(\lambda x.q^*((b_q^\delta)^\dagger(a_q^{\varepsilon_x,\delta})))\\ &\overset{\Delta \text{ def.}}{=} v(\lambda x.q^*(\Delta_x(q)))\\ &\overset{b \text{ def.}}{=} v(\lambda x.q^*(b_q^\Delta(x)))\\ &\overset{a \text{ def.}}{=} a_q^{v,\Delta}.\end{aligned}$$

⊣

It follows that the product operation of the monad J_R^T can be explicitly described in terms of the product of the monad T:

Lemma 2.3. *For all* $\varepsilon\colon (X \to R) \to TX$ *and* $\delta\colon X \to (Y \to R) \to TY$ *and* $q\colon X \times Y \to R$ *we have*

$$(\varepsilon \otimes^{J_R^T} \delta)(q) = a \otimes^T f,$$

where $f(x) \overset{TY}{=} \delta_x(q_x)$ *and* $a \overset{TX}{=} \varepsilon(\lambda x^X.(q_x)^*(fx))$.

Proof. Let $\Delta_x :\equiv \lambda y.\lambda p.\eta^T_{X\times Y}(x,y)$ and $\Theta :\equiv \lambda x.\lambda p.(b_p^{\Delta_x})^\dagger(a_p^{\delta_x,\Delta_x})$, where $a_{(\cdot)}^{(\cdot),(\cdot)}$ and $b_{(\cdot)}^{(\cdot)}$ refer to the constructions defined in Lemma 2.2. We will make use of the simple equation $q_x = q^* \circ (\lambda y.\eta^T_{X\times Y}(x,y))$. First note the following three facts:

$$(*)\quad b_q^{\Delta_x}(y) \overset{b\text{ def.}}{=} \Delta_x(y)(q) \overset{\Delta_x\text{ def.}}{=} \eta^T_{X\times Y}(x,y)$$

and

$$(+)\quad a_q^{\delta_x,\Delta_x} \overset{a\text{ def.}}{=} \delta_x(q^*\circ b_q^{\Delta_x}) \overset{(*)}{=} \delta_x(q^* \circ (\lambda y.\eta^T_{X\times Y}(x,y))) = \delta_x(q_x)$$

and

$$\begin{aligned}
(\dagger)\quad a &\overset{a\text{ def.}}{=} \varepsilon(\lambda x.(q_x)^*(fx))\\
&\overset{f\text{ def.}}{=} \varepsilon(\lambda x.(q_x)^*(\delta_x(q_x)))\\
&= \varepsilon(\lambda x.q^*((\lambda y.\eta^T_{X\times Y}(x,y))^\dagger(\delta_x(q_x))))\\
&\overset{(*)}{=} \varepsilon(\lambda x.q^*((b_q^{\Delta_x})^\dagger(\delta_x(q_x))))\\
&\overset{(+)}{=} \varepsilon(\lambda x.q^*((b_q^{\Delta_x})^\dagger(a_q^{\delta_x,\Delta_x})))\\
&\overset{\Theta\text{ def.}}{=} \varepsilon(\lambda x.q^*(\Theta(x)(q)))\\
&= \varepsilon(q^*\circ \lambda x.\Theta(x)(q))\\
&= \varepsilon(q^* \circ b_q^\Theta)\\
&\overset{a\text{ def.}}{=} a_q^{\varepsilon,\Theta}.
\end{aligned}$$

It then follows that:

$$\begin{aligned}
(\varepsilon \otimes^{J_R^T} \delta)(q) &\overset{\text{D1.7}}{=} (\lambda x.(\lambda y.\eta^{J_R^T}_{X\times Y}(x,y))^\dagger(\delta_x))^\dagger(\varepsilon)(q)\\
&\overset{\text{L2.2(i)}}{=} (\lambda x.(\Delta_x)^\dagger(\delta_x))^\dagger(\varepsilon)(q)\\
&\overset{\text{L2.2(ii)}}{=} (\lambda x.\lambda p.(b_p^{\Delta_x})^\dagger(a_p^{\delta_x,\Delta_x}))^\dagger(\varepsilon)(q)\\
&\overset{\Theta\text{ def.}}{=} \Theta^\dagger(\varepsilon)(q)\\
&\overset{\text{L2.2(ii)}}{=} (b_q^\Theta)^\dagger(a_q^{\varepsilon,\Theta})\\
&\overset{b\text{ def.}}{=} (\lambda x.(b_q^{\Delta_x})^\dagger(a_q^{\delta_x,\Delta_x}))^\dagger(a_q^{\varepsilon,\Theta})\\
&\overset{(*)}{=} (\lambda x.(\lambda y.\eta^T_{X\times Y}(x,y))^\dagger(a_q^{\delta_x,\Delta_x}))^\dagger(a_q^{\varepsilon,\Theta})\\
&\overset{(+)}{=} (\lambda x.(\lambda y.\eta^T_{X\times Y}(x,y))^\dagger(\delta_x(q_x)))^\dagger(a_q^{\varepsilon,\Theta})\\
&\overset{(\dagger)}{=} (\lambda x.(\lambda y.\eta^T_{X\times Y}(x,y))^\dagger(\delta_x(q_x)))^\dagger(a)\\
&\overset{f\text{ def.}}{=} (\lambda x.(\lambda y.\eta^T_{X\times Y}(x,y))^\dagger(fx))^\dagger(a)\\
&\overset{a\text{ def.}}{=} a \otimes^T f.
\end{aligned}$$

⊣

Definition 2.4 (From J_R^T to K_R). Let $K_R X = (X \to R) \to R$. Given a T-selection function $\varepsilon\colon J_R^T X$ we can construct a quantifier $\overline{\varepsilon}\colon K_R X$ as

$$\overline{\varepsilon}(p^{X \to R}) \overset{R}{=} p^*(\varepsilon p).$$

Recall that we are assuming R is a T-algebra.

It can be shown that the construction $\varepsilon \mapsto \overline{\varepsilon}$ is actually a monad morphism, from which the next lemma follows. Nevertheless, we shall prove the lemma directly. A particular instance of this lemma, when T is the identity monad, was first proven in [5]. It is important that R is a T-algebra.

Lemma 2.5. *Given* $\varepsilon \colon J_R^T X$ *and* $\delta \colon X \to J_R^T Y$ *then*

$$\overline{(\varepsilon \otimes^{J_R^T} \delta)} = \overline{\varepsilon} \otimes^{K_R} (\lambda x.\overline{\delta_x}).$$

Proof. Define $f(x) = \delta_x(q_x)$ and $p(x) = (q_x)^*(fx)$ and $a = \varepsilon(p)$. We calculate as follows:

$$\begin{aligned}
\overline{(\varepsilon \otimes^{J_R^T} \delta)}(q) &\overset{\text{D2.4}}{=} q^*((\varepsilon \otimes^{J_R^T} \delta)(q)) \\
&\overset{\text{L2.3}}{=} q^*(a \otimes^T f) \\
&\overset{\text{L1.9}}{=} (\lambda x.(q_x)^*(fx))^*(a) \\
&\overset{a \text{ def.}}{=} (\lambda x.(q_x)^*(fx))^*(\varepsilon(p)) \\
&\overset{p \text{ def.}}{=} p^*(\varepsilon(p)) \\
&\overset{\text{D2.4}}{=} \overline{\varepsilon}(p) \\
&\overset{p,f \text{ def.}}{=} \overline{\varepsilon}(\lambda x.(q_x)^*(\delta_x(q_x))) \\
&\overset{\text{D2.4}}{=} \overline{\varepsilon}(\lambda x.\overline{\delta_x}(q_x)) \\
&= (\overline{\varepsilon} \otimes^{K_R} (\lambda x.\overline{\delta_x}))(q).
\end{aligned}$$

The last equality in the chain above uses the definition of the product $\otimes$ for the strong monad $K_R X$. ⊣

§3. **Iterated products and bar recursion.** Given any strong monad M we can consider the particular case of the product operation $MZ \times (Z \to MY) \to M(Z \times Y)$ where $Z = X^*$ and $Y = X$, so as to obtain a mapping

$$M(X^*) \times (X^* \to MX) \to M(X^* \times X).$$

Composing with the mapping $M(X^* \times X) \to M(X^*)$, we have an operation which can be iterated

$$M(X^*) \times (X^* \to MX) \to M(X^*)$$

starting from the canonical element of type $M(X^*)$, leading to a functional[2] of type $(X^* \to M(X)) \to M(X^*)$. Although this will not be a total operation in general, it is surprising that, as shown in [5], it defines a total operation when M is the selection monad $MX = J_R X$ and R is a discrete type.

It is also possible to iterate the binary product of M in a controlled way, by using an explicit termination function $\omega \colon X^{\mathbb{N}} \to \mathbb{N}$ as

[2]A simpler instance of this operation without dependent types, namely $(MX)^{\mathbb{N}} \to M(X^{\mathbb{N}})$, is actually a built-in function in standard implementations of the Haskell programming language called sequence :: Monad m => [m a] -> m [a].

$$M\text{-EP}^{\omega}_{s}(\alpha) \overset{M(X^*)}{=} \begin{cases} \eta^M(\langle\,\rangle) & \text{if } \omega(s^+) < |s|, \\ \alpha_s \otimes^M (\lambda x.M\text{-EP}^{\omega}_{s*x}(\alpha)) & \text{otherwise,} \end{cases}$$

where M-EP$^{\omega}_{s}$ is of type $(X^* \to M(X)) \to M(X^*)$. Recall that α_s is an abbreviation for $\alpha(s)$ as explained in Section 1.1. We use the acronym M-EP for the "explicitly controlled iterated product of the strong monad M".

The explicitly controlled product of selection functions EPS or quantifiers EPQ (cf. [7]) are particular cases when $MX = J_R X$ and $MX = K_R X$, this time for an arbitrary R, i.e., EPS $= J_R$-EP and EPQ $= K_R$-EP. In turn, these are primitively recursively equivalent to restricted Spector bar recursion and the general Spector bar recursion, respectively [7].

In this section we consider another instance where $MX = J^T_R X$, with T being a strong monad, i.e., J^T_R-EP which we shall call T-EPS.

Definition 3.1 (Iterated product of T-selection functions). Let $\varepsilon_s \colon J^T_R X_{|s|}$ and $s \colon X^*$ and $\omega \colon X^{\mathbb{N}} \to \mathbb{N}$. We define $T\text{-EPS}^{\omega}_{s}(\varepsilon) \colon J^T_R X^*$ as $T\text{-EPS}^{\omega}_{s} = J^T_R\text{-EP}^{\omega}_{s}$.

Unfolding the definition of the binary product, as in Lemma 2.2, and noticing that $\eta^{J^T_R}(\langle\,\rangle) = \lambda q.\eta^T(\langle\,\rangle)$, the equation above can be also written as

$$T\text{-EPS}^{\omega}_{s}(\varepsilon)(q) = \begin{cases} \eta^T(\langle\,\rangle) & \text{if } \omega(s^+) < |s|, \\ a \otimes^T f & \text{otherwise,} \end{cases} \tag{2}$$

where $a = \varepsilon_s(\lambda x.(q_x)^*(fx))$ and $f(x) = T\text{-EPS}^{\omega}_{s*x}(\varepsilon)(q_x)$.

Recall that EPQ is the explicitly controlled iterated product of quantifiers, i.e., EPQ $= K_R$-EP. EPQ satisfies the equation

$$\text{EPQ}^{\omega}_{s}(\phi) = \begin{cases} \lambda q.q(\langle\,\rangle) & \text{if } \omega(s^+) < |s|, \\ \phi_s \otimes^{K_R} \lambda x.\text{EPQ}^{\omega}_{s*x}(\phi) & \text{otherwise.} \end{cases}$$

Again, the definition of the binary product of quantifiers can unfolded, leading to the equivalent equation

$$\text{EPQ}^{\omega}_{s}(\phi)(q) = \begin{cases} q(\langle\,\rangle) & \text{if } \omega(s^+) < |s|, \\ \phi_s(\lambda x.\text{EPQ}^{\omega}_{s*x}(\phi)(q_x)) & \text{otherwise.} \end{cases} \tag{3}$$

As shown in [4], EPQ is equivalent over system T to Spector's bar recursion. The following lemma can be seen as a generalisation of Lemma 2.5.

Lemma 3.2. $\overline{T\text{-EPS}^{\omega}_{\langle\,\rangle}(\varepsilon)} = \text{EPQ}^{\omega}_{\langle\,\rangle}(\overline{\varepsilon})$.

Proof. Let ω and ε be fixed, and define

$$P(s) \;:\equiv\; \overline{T\text{-EPS}^{\omega}_{s}(\varepsilon)} = \text{EPQ}^{\omega}_{s}(\overline{\varepsilon}).$$

The proof goes by bar induction. First, assuming $\omega(s^+) < |s|$, we must show $P(s)$,

$$\begin{aligned} \text{EPQ}^{\omega}_{s}(\overline{\varepsilon})(q) &= q(\langle\,\rangle) \\ &\overset{\text{D1.6(i)}}{=} q^*(\eta^T(\langle\,\rangle)) \\ &\overset{\text{D3.1}}{=} q^*(T\text{-EPS}^{\omega}_{s}(\varepsilon)(q)) \\ &\overset{\text{D2.4}}{=} \overline{T\text{-EPS}^{\omega}_{s}(\varepsilon)}. \end{aligned}$$

In the second case, assuming $\omega(s^+) \geq |s|$ we must show that $P(s)$ follows from $\forall x P(s * x)$. Hence, assume (IH) $\overline{T\text{-EPS}^{\omega}_{s*x}(\varepsilon)} = \mathsf{EPQ}^{\omega}_{s*x}(\overline{\varepsilon})$, for all x, we have:

$$\begin{aligned}
\mathsf{EPQ}^{\omega}_{s}(\overline{\varepsilon})(q) &= (\overline{\varepsilon} \otimes^{K_R} (\lambda x.\mathsf{EPQ}^{\omega}_{s*x}(\overline{\varepsilon})))(q) \\
&\overset{\text{(IH)}}{=} (\overline{\varepsilon} \otimes^{K_R} (\lambda x.\overline{T\text{-EPS}^{\omega}_{s*x}(\varepsilon)}))(q) \\
&\overset{\text{L2.5}}{=} \overline{(\varepsilon \otimes^{J^T_R} (\lambda x.T\text{-EPS}^{\omega}_{s*x}(\varepsilon)))}(q) \\
&= \overline{T\text{-EPS}^{\omega}_{s}(\varepsilon)}(q). \qquad \dashv
\end{aligned}$$

It is well know that the product of selection functions of type (X, R) can be simulated by a product where R is restricted to $R = X^{\mathbb{N}}$ and $q \colon X^{\mathbb{N}} \to R$ is the identity function. In fact, one can think of Spector's restricted form of bar recursion [17] as the iterated product of these restricted selection functions. In terms of games, it corresponds to taking the outcome of the game to be the sequence of moves played. The actual outcome of the game can be reconstructed from this sequence via the outcome function. The next lemma shows that this simulation of an arbitrary outcome type R by taking the outcome to be the actual sequence of moves also works in this monadic setting.

Lemma 3.3. *T-EPS of type (X, R) is definable from T-EPS of type $(X, TX^{\mathbb{N}})$.*

Proof. Let $\mathsf{add}_s \colon X^{\mathbb{N}} \to X^{\mathbb{N}}$ and $\mathsf{drop}_n \colon X^{\mathbb{N}} \to X^{\mathbb{N}}$ be the functions that append the finite sequence s to the beginning of an infinite list, and the function that drops n elements from an infinite list, respectively. Clearly, $\mathsf{drop}_{|s|} \circ \mathsf{add}_s$ is the identity, and hence, by functoriality, $T(\mathsf{drop}_{|s|}) \circ T(\mathsf{add}_s)$ is the identity on $T(X^{\mathbb{N}})$. Given $q \colon X^{\mathbb{N}} \to R$ and $\varepsilon_s \colon J^T_R X$ we define $\varepsilon^q_s \colon J^T_{TX^{\mathbb{N}}} X$ as

$$\varepsilon^q_s(p^{X \to TX^{\mathbb{N}}}) \overset{TX}{=} \varepsilon_s(\lambda x.((q_{s*x})^* \circ T(\mathsf{drop}_{|s*x|}))(px)).$$

Note that $TX^{\mathbb{N}}$ is also a T-algebra with the map

$$(\cdot)^* \colon (Y \to TX^{\mathbb{N}}) \to (TY \to TX^{\mathbb{N}})$$

being simply the $(\cdot)^\dagger$ of the monad T. We claim that

$$T\text{-EPS}^{\omega}_{\langle\rangle}(\varepsilon)(q) = T\text{-EPS}^{\omega}_{\langle\rangle}(\varepsilon^q)(\eta^T).$$

Define

$$P(s) \;:\equiv\; T\text{-EPS}^{\omega}_{s}(\varepsilon)(q_s) = T\text{-EPS}^{\omega}_{s}(\varepsilon^q)(\eta^T \circ \mathsf{add}_s)$$

and let us show $P(\langle\rangle)$ by bar induction. Recall that $T(\mathsf{add}_s) = \eta^T \circ \mathsf{add}_s$ by definition. In the base case, assuming $\omega(s^+) < |s|$, we have

$$T\text{-EPS}^{\omega}_{s}(\varepsilon)(q_s) = \eta^T(\langle\rangle) = T\text{-EPS}^{\omega}_{s}(\varepsilon^q)(\eta^T \circ \mathsf{add}_s).$$

For the bar inductive step we assume $P(s * x)$ holds for all x and must prove $P(s)$. We can also assume that $\omega(s^+) \geq |s|$. Let

$$\begin{aligned}
f(x) &= T\text{-EPS}^{\omega}_{s*x}(\varepsilon)(q_{s*x}) \\
a &= \varepsilon_s(\lambda x.(q_{s*x})^*(fx)) \\
\tilde{f}(x) &= T\text{-EPS}^{\omega}_{s*x}(\varepsilon^q)(\eta^T \circ \mathsf{add}_{s*x}) \\
\tilde{a} &= \varepsilon^q_s(\lambda x.T(\mathsf{add}_{s*x})(\tilde{f}x)).
\end{aligned}$$

By the bar inductive hypothesis we have $f = \tilde{f}$ and hence

$$\begin{aligned}\tilde{a} &= \varepsilon_s^q(\lambda x.T(\mathsf{add}_{s*x})(\tilde{f}x)) \\ &\overset{\text{(IH)}}{=} \varepsilon_s^q(\lambda x.T(\mathsf{add}_{s*x})(fx)) \\ &\overset{\varepsilon^q \text{ def.}}{=} \varepsilon_s(\lambda x.((q_{s*x})^* \circ T(\mathsf{drop}_{|s*x|}))(T(\mathsf{add}_{s*x})(fx))) \\ &= \varepsilon_s(\lambda x.(q_{s*x})^*(fx)) \\ &= a.\end{aligned}$$

Therefore

$$\begin{aligned}T\text{-}\mathsf{EPS}_s^\omega(\varepsilon)(q_s) &= a \otimes^T f \\ &= \tilde{a} \otimes^T \tilde{f} \\ &= T\text{-}\mathsf{EPS}_s^\omega(\varepsilon^q)(\eta^T \circ \mathsf{add}_s).\end{aligned}$$

In the last step we have used that $T(\mathsf{add}_s)$ is defined as $\eta^T \circ \mathsf{add}_s$. ⊣

The main result in this section is that Spector's original bar recursion already defines the explicitly controlled product of T-selection functions T-EPS. Spector proves this in [17] for the case when T is the identity monad. The following theorem shows that this in fact holds for any strong monad T.

THEOREM 3.4. *T-EPS is definable from* EPQ.

PROOF. We claim that $T\text{-}\mathsf{EPS}_{\langle\rangle}^\omega(\varepsilon)(q)$ can be defined as $\mathsf{EPQ}_{\langle\rangle}^\omega(\overline{\varepsilon^q})(\eta)$, where ε^q is as in the proof of the previous lemma. Indeed we have:

$$\begin{aligned}\mathsf{EPQ}_{\langle\rangle}^\omega(\overline{\varepsilon^q})(\eta) &\overset{\text{L3.2}}{=} \overline{T\text{-}\mathsf{EPS}_{\langle\rangle}^\omega(\varepsilon^q)}(\eta) \\ &\overset{\text{D2.4}}{=} \eta^*(T\text{-}\mathsf{EPS}_{\langle\rangle}^\omega(\varepsilon^q)(\eta)) \\ &\overset{\text{L3.3}}{=} \eta^*(T\text{-}\mathsf{EPS}_{\langle\rangle}^\omega(\varepsilon)(q)) \\ &= \eta^\dagger(T\text{-}\mathsf{EPS}_{\langle\rangle}^\omega(\varepsilon)(q)) \\ &\overset{\text{D1.3(i)}}{=} T\text{-}\mathsf{EPS}_{\langle\rangle}^\omega(\varepsilon)(q).\end{aligned}$$

We used that the map $(\cdot)^*$ for the algebra $TX^{\mathbb{N}}$ is just the $(\cdot)^\dagger$ map for the monad T, as discussed in the proof of Lemma 3.3. ⊣

§4. Finite powersets.

For the rest of the paper we will make essential use of the definitional extension of Gödel's system T with the finite powerset monad $\mathcal{P}_{\mathrm{f}}(X)$, already discussed in Examples 1.5 and 1.8. For the rest of the paper we shall assume that $R = \mathcal{P}_{\mathrm{f}}(R')$, for some R', so that R is an algebra for $\mathcal{P}_{\mathrm{f}}(\cdot)$ with $(\cdot)^* = (\cdot)^\dagger$. We will also use $\bigcup\colon \mathcal{P}_{\mathrm{f}}(R) \to R$, the usual union operation which satisfies $S_i \subseteq \bigcup\{S_i : i \in I\}$ (we use this in Lemma 4.5).

DEFINITION 4.1 (Herbrand bar recursion). Let us write hBR for the instance of T-EPS where $T = \mathcal{P}_{\mathrm{f}}(\cdot)$, i.e.,

$$\mathsf{hBR}_s^\omega(\varepsilon)(q) = \begin{cases}\{\langle\rangle\} & \text{if } \omega(s^+) < |s|, \\ \{a * r : a \in \chi \wedge r \in \mathsf{hBR}_{s*a}^\omega(\varepsilon)(q_a)\} & \text{otherwise,}\end{cases}$$

where $\chi = \varepsilon_s(\lambda x.\bigcup\{q_x(r) : r \in \mathsf{hBR}_{s*x}^\omega(\varepsilon)(q_x)\})$.

By Theorem 3.4 hBR is T-definable from Spector's general form of bar recursion [15]. We now prove four lemmas about hBR, to be used in the interpretation of DNS in the following section. For this section we will assume that ε and ω are fixed functionals and hence, for the sake of readability, we shall omit these as parameters in $\mathsf{hBR}^{\omega}_{s}(\varepsilon)(q)$.

Lemma 4.2. *Let $t = \mathsf{hBR}_{\langle\,\rangle}(q)$ and $s \in t$. For all $i \leq |s|$ we have*

$$s \in \{\langle s_0, \dots, s_{i-1}\rangle * r \ :\ r \in \mathsf{hBR}_{\langle s_0,\dots,s_{i-1}\rangle}(q_{\langle s_0,\dots,s_{i-1}\rangle})\}.$$

The types are $t\colon \mathcal{P}_{\mathrm{f}}(X^*)$ and $s\colon X^*$.

Proof. By induction on i. If $i = 0$ then $\langle s_0, \dots, s_{i-1}\rangle$ is the empty sequence and the result follows by the assumption that $s \in t$. For the induction step assume that $i < |s|$ and that

$$s \in \{\langle s_0, \dots, s_{i-1}\rangle * r \ :\ r \in \mathsf{hBR}_{\langle s_0,\dots,s_{i-1}\rangle}(q_{\langle s_0,\dots,s_{i-1}\rangle})\}.$$

Since $i < |s|$ there must exist some $r \in \mathsf{hBR}_{\langle s_0,\dots,s_{i-1}\rangle}(q_{\langle s_0,\dots,s_{i-1}\rangle})$ of the form $s_i * r'$ so that

(i) $s = \langle s_0, \dots, s_{i-1}, s_i\rangle * r'$, and

(ii) $s_i * r' \in \mathsf{hBR}_{\langle s_0,\dots,s_{i-1}\rangle}(q_{\langle s_0,\dots,s_{i-1}\rangle})$.

In particular, we cannot have $\mathsf{hBR}_{\langle s_0,\dots,s_{i-1}\rangle}(q_{\langle s_0,\dots,s_{i-1}\rangle}) = \{\langle\,\rangle\}$, so it must be the case that $(*)$ $\omega(\langle s_0, \dots, s_{i-1}\rangle^+) \geq |\langle s_0, \dots, s_{i-1}\rangle|$. Hence

$$\mathsf{hBR}_{\langle s_0,\dots,s_{i-1}\rangle}(q_{\langle s_0,\dots,s_{i-1}\rangle}) = \{a * r \ :\ a \in \chi \wedge r \in \mathsf{hBR}_{\langle s_0,\dots,s_{i-1},a\rangle}(q_{\langle s_0,\dots,s_{i-1},a\rangle})\},$$

where

$$\chi = \varepsilon_{\langle s_0,\dots,s_{i-1}\rangle}(\lambda y^X . \bigcup\{q_{\langle s_0,\dots,s_{i-1},y\rangle}(r) \ :\ r \in \mathsf{hBR}_{\langle s_0,\dots,s_{i-1},y\rangle}(q_{\langle s_0,\dots,s_{i-1},y\rangle})\}).$$

From (ii) it follows that $s_i \in \chi$ and

(iii) $r' \in \mathsf{hBR}_{\langle s_0,\dots,s_{i-1},s_i\rangle}(q_{\langle s_0,\dots,s_{i-1},s_i\rangle})$.

Finally, from (i) and (iii) we have

$$s \in \{\langle s_0, \dots, s_{i-1}, s_i\rangle * r' \ :\ r' \in \mathsf{hBR}_{\langle s_0,\dots,s_{i-1},s_i\rangle}(q_{\langle s_0,\dots,s_{i-1},s_i\rangle})\},$$

which concludes the proof. $\dashv$

For the following three lemmas let $t = \mathsf{hBR}_{\langle\,\rangle}(q)$, and assume $a_0, \dots, a_n$ is a finite sequence satisfying, for all $i \leq n$,

$$a_i \in \varepsilon_{\langle a_0,\dots,a_{i-1}\rangle}(p_i),$$

where p_i is defined as

$$p_i(y) = \bigcup\{q_{\langle a_0,\dots,a_{i-1},y\rangle}(r) \ :\ r \in \mathsf{hBR}_{\langle a_0,\dots,a_{i-1},y\rangle}(q_{\langle a_0,\dots,a_{i-1},y\rangle})\}.$$

Lemma 4.3. *If $\omega(\langle a_0, \dots, a_{i-1}\rangle^+) \geq i$, for all $i \leq n$, then*

$$\langle a_0, \dots, a_{n-1}\rangle * x * r \in t,$$

for all $x \in \varepsilon_{\langle a_0,\dots,a_{n-1}\rangle}(p_n)$ and $r \in \mathsf{hBR}_{\langle a_0,\dots,a_{n-1},x\rangle}(q_{\langle a_0,\dots,a_{n-1},x\rangle})$.

Proof. We prove the lemma by induction on n.

For $n = 0$ the assumption of the lemma always holds, while the conclusion follows by the definition of hBR

$$t = \mathsf{hBR}_{\langle\,\rangle}(q) = \{a * r \ :\ a \in \varepsilon_{\langle\,\rangle}(p_0) \wedge r \in \mathsf{hBR}_a(q_a)\}$$

since $\omega(\langle\,\rangle^+) \geq 0$. For the induction step, assume that $\omega(\langle a_0, \dots, a_{i-1}\rangle^+) \geq i$, for all $i \leq n+1$. In particular this holds for $i \leq n$. Hence, by induction hypothesis we have

$$\langle a_0, \dots, a_{n-1}\rangle * x * r \in t,$$
$$\text{for all } x \in \varepsilon_{\langle a_0,\dots,a_{n-1}\rangle}(p_n) \text{ and } r \in \mathsf{hBR}_{\langle a_0,\dots,a_{n-1},x\rangle}(q_{\langle a_0,\dots,a_{n-1},x\rangle}),$$

and, since $a_n \in \varepsilon_{\langle a_0,\dots,a_{n-1}\rangle}(p_n)$,

(i) $\langle a_0, \dots, a_n\rangle * r \in t$, for all $r \in \mathsf{hBR}_{\langle a_0,\dots,a_n\rangle}(q_{\langle a_0,\dots,a_n\rangle})$.

Now fix a $y \in \varepsilon_{\langle a_0,\dots,a_n\rangle}(p_{n+1})$ and an $r' \in \mathsf{hBR}_{\langle a_0,\dots,a_n,y\rangle}(q_{\langle a_0,\dots,a_n,y\rangle})$. In order to show that $\langle a_0, \dots, a_n\rangle * y * r' \in t$, by (i) it is enough to show that $y * r' \in \mathsf{hBR}_{\langle a_0,\dots,a_n\rangle}(q_{\langle a_0,\dots,a_n\rangle})$. But since $\omega(\langle a_0, \dots, a_n\rangle^+) \geq n+1$, this indeed follows by the definition of hBR, and the assumptions on y and r'. ⊣

Lemma 4.4. *Let t and a_i's be as above. Define $N = 1 + \max\{|s| \ : \ s \in t\}$. For some $n < N$ we have that*

(a) *n is the least such that $\omega(\langle a_0, \dots, a_n\rangle^+) < n+1$, and*
(b) *$\langle a_0, \dots, a_n\rangle \in t$.*

Proof. Suppose that for all $n \leq N$ we have $\omega(\langle a_0, \dots, a_{n-1}\rangle^+) \geq n$. By Lemma 4.3 this would imply $\langle a_0, \dots, a_{N-1}\rangle * r \in t$ for some nonempty finite sequence r, which is a contradiction by the definition of N. Therefore, let $n < N$ be the smallest such that $\omega(\langle a_0, \dots, a_n\rangle^+) < n+1$, so that for all $i \leq n$ we have $\omega(\langle a_0, \dots, a_{i-1}\rangle^+) \geq i$. By Lemma 4.3 again we have that $\langle a_0, \dots, a_{n-1}\rangle * a_n * r \in t$ for all $r \in \mathsf{hBR}_{\langle a_0,\dots,a_n\rangle}(q_{\langle a_0,\dots,a_n\rangle})$. But since $\omega(\langle a_0, \dots, a_n\rangle^+) < n+1$ we have that $\mathsf{hBR}_{\langle a_0,\dots,a_n\rangle}(q_{\langle a_0,\dots,a_n\rangle}) = \{\langle\,\rangle\}$, implying $\langle a_0, \dots, a_n\rangle \in t$. ⊣

Lemma 4.5. *Let t, p_i, a_i be as above, and $n < N$ as in Lemma 4.4. Let also $s = \langle a_0, \dots, a_n\rangle$. Then for all $i \leq n$*

$$q(s) \ \subseteq \ p_i(a_i). \tag{4}$$

Proof. By Lemma 4.4 we have that $s \in t$. Hence, by Lemma 4.2, for $i \leq n$

$$s \in \{\langle a_0, \dots, a_{i-1}, a_i\rangle * r \ : \ r \in \mathsf{hBR}_{\langle a_0,\dots,a_{i-1},a_i\rangle}(q_{\langle a_0,\dots,a_{i-1},a_i\rangle})\}.$$

It follows that

$$q(s) \in \{q(\langle a_0, \dots, a_{i-1}, a_i\rangle * r) \ : \ r \in \mathsf{hBR}_{\langle a_0,\dots,a_{i-1},a_i\rangle}(q_{\langle a_0,\dots,a_{i-1},a_i\rangle})\}.$$

Hence

$$\begin{aligned} q(s) \ &\subseteq \ \bigcup\{q_{\langle a_0,\dots,a_{i-1},a_i\rangle}(r) \ : \ r \in \mathsf{hBR}_{\langle a_0,\dots,a_{i-1},a_i\rangle}(q_{\langle a_0,\dots,a_{i-1},a_i\rangle})\} \\ &= \ p_i(a_i), \end{aligned}$$

which concludes the proof. ⊣

§5. Application: Herbrand interpretation of DNS.

In this final section we obtain an extension of the Herbrand functional interpretation of nonstandard arithmetic [19] to include an interpretation of the double negation shift

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

where $\forall^{\mathsf{st}} x A(x)$ is the quantification over standard objects. We will show that the product of T-selection functions, with T being the finite powerset monad, witnesses the Herbrand functional interpretation of DNS. Let us first briefly recall here the definition of the Herbrand functional interpretation from [19]. We shall only present the $\{\to, \forall^{\mathsf{st}}, \bot\}$-fragment as this is enough to carry out the interpretation of DNS. Negation $\neg A$ is defined as $A \to \bot$. Although we will present an explicit definition for the witnesses of DNS, for simplicity, we will carry out the *verification of correctness* in a classical setting, reading the weak existential $\neg\forall x \neg A$ as the strong one $\exists x A$.

To simplify the exposition, we will also abbreviate $\mathcal{P}_{\mathrm{f}}(X \to Y)$ as $X \Rightarrow Y$, i.e., the type of finite sets of functions from X to Y. We can think of the elements $f : X \Rightarrow \mathcal{P}_{\mathrm{f}}(Y)$ as functions by defining the following *set-application*[3]

$$\mathrm{Ap}(f)(x^X) \overset{\mathcal{P}_{\mathrm{f}}(Y)}{=} \bigcup_{g \in f} gx.$$

Hence, if $f : X \Rightarrow \mathcal{P}_{\mathrm{f}}(Y)$ then $\mathrm{Ap}(f)(\cdot) : X \to \mathcal{P}_{\mathrm{f}}(Y)$. In particular, if $f : (X \Rightarrow (Y \Rightarrow \mathcal{P}_{\mathrm{f}}(Z)))$ then $\mathrm{Ap}(\mathrm{Ap}(f)(x))(y)$ stands for

$$\bigcup_{g \in f} \bigcup_{h \in gx} hy,$$

and we will be abbreviated that as $\mathrm{Ap}^2(f)(x, y)$.

Definition 5.1 ([19]). The Herbrand functional interpretation of a formula A is defined by structural induction. Assume[4] $(A)^H = \exists^{\mathsf{st}} a^X \forall^{\mathsf{st}} b^Y A_H(a, b)$ and $(B)^H = \exists^{\mathsf{st}} c^V \forall^{\mathsf{st}} d^W B_H(c, d)$. The only relevant cases for the interpretation of DNS are:

$$(\bot)^H \equiv \bot,$$

$$(A \to B)^H \equiv \exists^{\mathsf{st}} f, g \forall^{\mathsf{st}} a^X, d^W (\forall b \in \mathrm{Ap}^2(g)(a, d)\, A_H(a, b) \to B_H(\mathrm{Ap}(f)(a), d)),$$

$$(\forall^{\mathsf{st}} z^Z A)^H \equiv \exists^{\mathsf{st}} h^{Z \Rightarrow X} \forall^{\mathsf{st}} z, b A_H(\mathrm{Ap}(h)(z), b),$$

where in the clause for $A \to B$ the types of f and g are

$$f \ : X \Rightarrow V,$$

$$g \ : X \Rightarrow (W \Rightarrow \mathcal{P}_{\mathrm{f}}(Y)).$$

For all other cases, including the other base cases, see [19].

Let us start by working out the Herbrand interpretation of negation $\neg A$ and double-negation $\neg\neg A$. If $(A)^H = \exists^{\mathsf{st}} a^X \forall^{\mathsf{st}} b^R A_H(a, b)$ then

$$(\neg A)^H \equiv \exists^{\mathsf{st}} p^{X \Rightarrow \mathcal{P}_{\mathrm{f}}(R)} \forall^{\mathsf{st}} a^X \neg \forall b \in \mathrm{Ap}(p)(a) A_H(a, b),$$

and hence

$$(\neg\neg A)^H \equiv \exists^{\mathsf{st}} \varepsilon \forall^{\mathsf{st}} p^{X \Rightarrow \mathcal{P}_{\mathrm{f}}(R)} \exists a^X \in \mathrm{Ap}(\varepsilon)(p) \forall b \in \mathrm{Ap}(p)(a) A_H(a, b),$$

where $\varepsilon : (X \Rightarrow \mathcal{P}_{\mathrm{f}}(R)) \Rightarrow \mathcal{P}_{\mathrm{f}}(X)$. Assuming $A(n)$ has a Herbrand functional interpretation $\exists^{\mathsf{st}} a^X \forall^{\mathsf{st}} b^R A_H(n, a, b)$ then the interpretation of $\forall^{\mathsf{st}} n^{\mathbb{N}} \neg\neg A(n)$ is

$$\exists^{\mathsf{st}} \delta \forall^{\mathsf{st}} n \forall^{\mathsf{st}} p^{X \Rightarrow \mathcal{P}_{\mathrm{f}}(R)} \exists a \in \mathrm{Ap}^2(\delta)(n, p) \forall b \in \mathrm{Ap}(p)(a) A_H(n, a, b). \tag{5}$$

[3] As pointed out by the referee, note that $W \Rightarrow (X \Rightarrow Y)$ and $X \Rightarrow (W \Rightarrow Y)$ are not canonically isomorphic: If W, X and Y are finite sets with cardinalities k, n and m respectively, then $W \Rightarrow (X \Rightarrow Y)$ has $2^{2^{km^n}}$ elements, whereas $X \Rightarrow (W \Rightarrow Y)$ has $2^{2^{nm^k}}$ elements. This lack of canonicity, however, is not an obstacle for the soundness of the Herbrand functional interpretation.

[4] Here a, b, c and d are potentially tuples of variables, though for simplicity we will treat them as if they were single variables.

The interpretation of the conclusion of DNS, $\neg\neg\forall^{\mathsf{st}} n^{\mathbb{N}} A(n)$, follows from[5]

$$\exists^{\mathsf{st}}\alpha\forall^{\mathsf{st}}\varphi, q\exists\beta \in \mathrm{Ap}^2(\alpha)(\varphi, q)\forall n \in \mathrm{Ap}(\varphi)(\beta)\forall b \in \mathrm{Ap}(q)(\beta)A_H(n, \beta(n), b), \quad (6)$$

where the types above are

$$\begin{aligned} &\delta\colon \mathbb{N} \Rightarrow (X \Rightarrow \mathcal{P}_{\mathrm{f}}(R)) \Rightarrow \mathcal{P}_{\mathrm{f}}(X), && p\colon X \Rightarrow \mathcal{P}_{\mathrm{f}}(R),\\ &q\colon (\mathbb{N} \to X) \Rightarrow \mathcal{P}_{\mathrm{f}}(R), && \beta\colon \mathbb{N} \to X,\\ &\varphi\colon (\mathbb{N} \to X) \Rightarrow \mathcal{P}_{\mathrm{f}}(\mathbb{N}), && \mathrm{Ap}^2(\alpha)(\varphi, q)\colon \mathcal{P}_{\mathrm{f}}(\mathbb{N} \Rightarrow X).\end{aligned}$$

Given δ, φ, and q, we will calculate finite sets α, N and P and show that

$$\begin{aligned}&\forall n \in N \forall p \in P \exists a \in \mathrm{Ap}^2(\delta)(n, p)\forall b \in \mathrm{Ap}(p)(a) A_H(n, a, b)\\ &\qquad\to \exists\beta \in \alpha\forall n \in \mathrm{Ap}(\varphi)(\beta)\forall b \in \mathrm{Ap}(q)(\beta)A_H(n, \beta(n), b).\end{aligned}$$

Although the Herbrand interpretation here would only actually ask us to produce finite sets of candidate "constructions" for α, N and P, with a guarantee that one of them did the job, we show that in fact we can produce concrete finite sets α, N, and P. Given δ, φ, and q as above, let us define

$$\begin{aligned}&\varepsilon_n\colon (X \to \mathcal{P}_{\mathrm{f}}(R)) \to \mathcal{P}_{\mathrm{f}}(X),\\ &\hat{q}\colon X^* \to \mathcal{P}_{\mathrm{f}}(R),\\ &\omega\colon (\mathbb{N} \to X) \to \mathbb{N}\end{aligned}$$

as

$$\begin{aligned}\varepsilon_n(p) &= \mathrm{Ap}^2(\delta)(n, \{p\}),\\ \hat{q}(s) &= \mathrm{Ap}(q)(s^+),\\ \omega(\beta) &= \max(\mathrm{Ap}(\varphi)(\beta)).\end{aligned}$$

We will then apply hBR to ε_n, $\hat{q}$, and ω.

THEOREM 5.2. *Define* $t = \mathsf{hBR}^{\omega}_{\langle\rangle}(\varepsilon)(\hat{q})$. *We claim that*

$$\begin{aligned}\alpha &= \{s^+ : s \in t\},\\ P &= \{p_r : r \preceq s \wedge s \in t\},\\ N &= 1 + \max\{|s| : s \in t\},\end{aligned}$$

where $p_r(y) = \bigcup\{\hat{q}(r * y * r') : r' \in \mathsf{hBR}(r * y)\}$, *witness the Herbrand interpretation of* DNS, *i.e.*,

$$\begin{aligned}&\forall n \le N \forall p \in P \exists a \in \mathrm{Ap}^2(\delta)(n, p)\forall b \in \mathrm{Ap}(p)(a) A_H(n, a, b)\\ &\qquad\to \exists\beta \in \alpha\forall i \in \mathrm{Ap}(\varphi)(\beta)\forall b \in \mathrm{Ap}(q)(\beta)A_H(i, \beta(i), b)\end{aligned}$$

viewing the number N *as the finite set* $\{0, 1, \ldots, N\}$.

PROOF. Assume

$$\forall n \le N \forall p \in P \exists a \in \mathrm{Ap}^2(\delta)(n, p)\forall b \in \mathrm{Ap}(p)(a) A_H(n, a, b). \quad (7)$$

[5]Instead of producing a set of functions $\beta\colon \mathbb{N} \Rightarrow X$ we will actually produce a single function $\beta\colon \mathbb{N} \to X$. Note that $\mathrm{Ap}(\{p\})(a) = p(a)$.

By induction on n it follows that: For all $n \leq N$ there exists a sequence $\langle a_0, \ldots, a_n \rangle$ such that either

- for some $i < n$, $\omega(\langle a_0, \ldots, a_i \rangle^+) < i + 1$, or
- for all $i \leq n$,

$$a_i \in \underbrace{\mathrm{Ap}^2(\delta)(i, \{p_{\langle a_0,\ldots,a_{i-1}\rangle}\})}_{\varepsilon_i(p_{\langle a_0,\ldots,a_{i-1}\rangle})} \wedge \forall b \in \underbrace{\mathrm{Ap}(\{p_{\langle a_0,\ldots,a_{i-1}\rangle}\})(a_i)}_{p_{\langle a_0,\ldots,a_{i-1}\rangle}(a_i)} A_H(i, a_i, b). \quad (8)$$

We have used Lemma 4.3, since under the assumption that $\omega(\langle a_0, \ldots, a_{i-1} \rangle^+) \geq i$ for all $i \leq n$ then $\langle a_0, \ldots, a_{i-1} \rangle * r \in t$, for some r, and hence $p_{\langle a_0,\ldots,a_{i-1}\rangle} \in P$. By Lemma 4.4 there exists a least $n < N$ such that $\omega(\langle a_0, \ldots, a_n \rangle^+) < n+1$, so that (8) holds for all $i \leq n$, and $\langle a_0, \ldots, a_n \rangle \in t$. Let $s = \langle a_0, \ldots, a_n \rangle$ and $\beta = s^+$ (so that $\beta \in \alpha$). Note that

$$\max(\mathrm{Ap}(\varphi)(s^+)) = \omega(s^+) < |s|.$$

Hence, $i < |s|$ for all $i \in \mathrm{Ap}(\varphi)(s^+)$. By Lemma 4.5

$$\mathrm{Ap}(q)(\beta) = \mathrm{Ap}(q)(s^+) = \hat{q}(s) \subseteq p_{\langle a_0,\ldots,a_{i-1}\rangle}(a_i), \text{ for all } i \in \mathrm{Ap}(\varphi)(s^+).$$

By (8) we can conclude that $\forall i \in \mathrm{Ap}(\varphi)(\beta) \forall b \in \mathrm{Ap}(q)(\beta) A_H(i, \beta(i), b)$. ⊣

Remark 5.3. We note that the theory of nonstandard numbers is not required for the interpretation of the double negation shift: The Herbrand functional interpretation of axioms of nonstandard analysis [19] is orthogonal to the interpretation of DNS. The important aspect of the Herbrand functional interpretation used here is that it witnesses some quantifiers via finite sets, rather than precise witnesses. This in particular means that one can also consider the Herbrand functional interpretation of a standard theory such as $\mathsf{HA}^\omega + \mathsf{AC}_0 + \mathsf{DNS}$.

§6. Further work.

A reader familiar with the bounded functional interpretation of DNS (cf. [8]) will have noticed several similarities with the Herbrand functional interpretation of DNS presented here. The main difference, however, is that we have made no effort to formalise the *verification* of the interpretation in a constructive setting, choosing to view $\neg\forall x \neg A$ as a strong existence $\exists x A$. Although it is clear to us that such formalisation is possible, attempting to do so would complicate the verification and probably obfuscate the crucial steps of the bar recursive construction. We hope that by simplifying the “logical component” of the proof one can better appreciate its “computational” aspect and the use of the “Herbrand” bar recursion. The recent paper [9] sheds some light at the relationship between the two interpretations.

Note also that all lemmas of Section 4 were proven for the specific case of the finite powerset monad only. It is reasonable to ask whether more general versions of such lemmas work already for the monadic bar recursion T-EPS. The main challenge as we see it is to find the appropriate abstraction to the notion of set containment and subset inclusion. Similarly, one might consider generalisations of the Herbrand functional interpretation whereby the finite powerset monads is replaced by an arbitrary monad, with possibly some extra structure.

§7. Acknowledgements. We would like to thank Benno van den Berg for bringing to our attention the open problem of giving a Herbrand functional interpretation for DNS. We would also like to thank the anonymous referee for detailed comments and suggestions.

REFERENCES

[1] S. Berardi, M. Bezem, and T. Coquand, *On the computational content of the axiom of choice*, this Journal, vol. 63 (1998), no. 2, pp. 600–622.

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

[3] M. Bezem, *Strongly majorizable functionals of finite type: A model for bar recursion containing discontinuous functionals*, this Journal, vol. 50 (1985), pp. 652–660.

[4] M. Escardó and P. Oliva, *Computational interpretations of analysis via products of selection functions*, ***Computability in Europe 2010*** (F. Ferreira, B. Lowe, E. Mayordomo, and L. M. Gomes, editors), Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2010, pp. 141–150.

[5] ———, *Selection functions, bar recursion, and backward induction*. ***Mathematical Structures in Computer Science***, vol. 20 (2010), no. 2, pp. 127–168.

[6] ———, *Sequential games and optimal strategies*. ***Royal Society Proceedings A***, vol. 467 (2011), pp. 1519–1545.

[7] ———, *Bar recursion and products of selection functions*, this Journal, vol. 80 (2015), no. 1, pp. 1–28.

[8] F. Ferreira and P. Engrácia, *The bounded functional interpretation of the double negation shift*, this Journal, vol. 75 (2010), no. 2, pp. 759–773.

[9] F. Ferreira and J. Gaspar, *Nonstandardness and the bounded functional interpretation*. ***Annals of Pure and Applied Logic***, vol. 166 (2015), pp. 665–740.

[10] K. Gödel, *Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes*. ***Dialectica***, vol. 12 (1958), pp. 280–287.

[11] A. Kock, *Strong functors and monoidal monads*. ***Archiv der Mathematik***, vol. 23 (1972), pp. 113–120.

[12] U. Kohlenbach, *Effective bounds from ineffective proofs in analysis: An application of functional interpretation and majorization*, this Journal, vol. 57 (1992), pp. 1239–1273.

[13] E. Moggi, *Notions of computation and monads*. ***Information and Computation***, vol. 1 (1991), pp. 55–92.

[14] A. Mostowski, *On a generalization of quantifiers*. ***Fundamenta Mathematicae***, vol. 44 (1957), pp. 12–36.

[15] P. Oliva and T. Powell, *On Spector's bar recursion*. ***Mathematical Logic Quarterly***, vol. 58 (2012), no. 4–5, pp. 356–365.

[16] B. Scarpellini, *A model for bar recursion of higher types*. ***Compositio Mathematica***, vol. 23 (1971), pp. 123–153.

[17] C. Spector, *Provably recursive functionals of analysis: A consistency proof of analysis by an extension of principles in current intuitionistic mathematics*, ***Recursive Function Theory: Proceedings of Symposia in Pure Mathematics, vol. 5*** (F. D. E. Dekker, editor), American Mathematical Society, Providence, Rhode Island, 1962, pp. 1–27.

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

[19] B. van den Berg, E. M. Briseid, and P. Safarik, *A functional interpretation for nonstandard arithmetic*. ***Annals of Pure and Applied Logic***, vol. 163 (2012), no. 2, pp. 1962–1994.

[20] P. Wadler, *The essence of functional programming*, ***Proceedings of the 19th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages***, POPL'92, ACM, New York, NY, USA, 1992, pp. 1–14.

SCHOOL OF ELECTRONIC ENGINEERING AND COMPUTER SCIENCE
QUEEN MARY UNIVERSITY OF LONDON
LONDON, UK
E-mail: p.oliva@qmul.ac.uk

SCHOOL OF COMPUTER SCIENCE
UNIVERSITY OF BIRMINGHAM
BIRMINGHAM, UK
E-mail: m.escardo@cs.bham.ac.uk