Selection Functions, Bar Recursion, and Backward Induction

Martín Escardó[1] and Paulo Oliva[2]

[1] *University of Birmingham, Birmingham B15 2TT, UK.*
[2] *Queen Mary University of London, London E1 4NS, UK*

Bar recursion arises in constructive mathematics, logic, proof theory and higher-type computability theory. We explain bar recursion in terms of sequential games, and show how it can be naturally understood as a generalisation of the principle of backward induction that arises in game theory. In summary, bar recursion calculates optimal plays and optimal strategies, which, for particular games of interest, amount to equilibria. We consider finite games and continuous countably infinite games, and relate the two. The above development is followed by a conceptual explanation of how the finite version of the main form of bar recursion considered here arises from a strong monad of selections functions that can be defined in any cartesian closed category. Finite bar recursion turns out to be a well-known morphism available in any strong monad, specialised to the selection monad.

1. Introduction

We define a generalisation of *sequential games* and investigate constructions of optimal outcomes and strategies via a form of bar recursion (Berardi et al., 1998; Berger and Oliva, 2006; Spector, 1962), which we propose as a formalisation of the principle of backward induction from game theory (Nisan et al., 2007). Our sequential games are defined in terms of rounds, where X_i are the possible moves at round i, leaving open the number of players, and who plays at each round. The outcome of a game is specified by an n-ary predicate $p \colon \Pi_{i=0}^{n-1} X_i \to R$, and the aim of the game by a quantifier for each round of the game. For instance, assume that R is the set $\mathbb{B} = \{\text{true}, \text{false}\}$ of booleans, and consider a game between two players, playing in alternating rounds, with the first player trying to force the outcome to be the value true while the second player tries to obtain the opposite outcome false. The first player has a winning strategy if and only if

$$\exists x_0 \forall x_1 \exists x_2 \forall x_3 \ldots p(x_0, \ldots, x_{n-1}).$$

On the other hand, assume the aim at each round is to force the outcome to be the value true. In this case, the existence of a winning strategy corresponds to the satisfiability of the predicate p, i.e.

$$\exists x_0 \exists x_1 \ldots \exists x_{n-1} p(x_0, \ldots, x_{n-1}).$$

Dually, if the goal of each round is to obtain a final outcome false, the non-existence of a winning strategy corresponds to the tautology of the predicate p, i.e.

$$\forall x_0 \forall x_1 \ldots \forall x_{n-1} p(x_0, \ldots, x_{n-1}).$$

Now consider games with more than two outcomes, e.g. $R = \{-1, 0, 1\}$. Following up from our first example, suppose the outcome 1 means that the first player wins, -1 means that the second player wins, and 0 stands for a draw. In this case, the existence of a non-losing strategy for the first player is expressed as

$$\left(\sup_{x_0 \in X_0} \inf_{x_1 \in X_1} \ldots \sup_{x_{n-2} \in X_{n-2}} \inf_{x_{n-1} \in X_{n-1}} p(x_0, \ldots, x_{n-1}) \right) \geq 0.$$

Similarly, if all inf functionals are replaced by sup, this corresponds to a game where each round is trying to maximise the final global payoff $p(x_0, \ldots, x_{n-1})$. In this last case, if $R = \mathbb{R}^n$ and at each round i we are trying to maximise the i-coordinate of the outcome, the existence of a winning strategy corresponds to the existence of a profile in Nash equilibrium for sequential games (Nisan et al., 2007).

In summary, the goal at each round i, in an n-round game, is defined via an *outcome quantifier*,

$$\phi_i \colon (X_i \to R) \to R,$$

which we leave open in the definition of the game. In the case ϕ_i are the standard quantifiers $\exists, \forall \colon (X \to \mathbb{B}) \to \mathbb{B}$ or the supremum and infimum functionals $\sup, \inf \colon (X \to R) \to R$, where R is a closed and bounded set of real numbers, we obtain the examples mentioned above. We then define the product of generalised quantifiers and use it to define notions such as optimal play, outcome and strategy.

Some generalised quantifiers $\phi \colon (X \to R) \to R$ have *selection functions*, i.e. functions

$$\varepsilon \colon (X \to R) \to X$$

satisfying $\phi(p) = p(\varepsilon(p))$. For example, a selection function for the supremum functional $\sup \colon (X \to R) \to R$, when it exists, gives a point at which p attains its maximum value $\max p$. We show that, when outcome quantifiers have selection functions, an optimal strategy for the game can be computed via a suitably defined product of corresponding selection functions. This product will turn out to appear not only in game theory (corresponding to backward induction (Nisan et al., 2007)), but also in algorithms (corresponding to backtracking (Valiente, 2002)) and proof theory (corresponding to bar recursion (Berardi et al., 1998; Berger and Oliva, 2006; Spector, 1962)), among others.

We then consider the infinite iteration of the binary product of selection functions, and discuss how this gives optimal strategies in finite games of unbounded length. Both the finite and infinite products considered here are generalisations of previous work of the first author (Escardó, 2008). This is explained in Section 5, where we also show how the infinite product amounts to a form of bar recursion.

The above development is followed by a conceptual explanation of how the finite version of the main form of bar recursion considered here arises from a strong monad of selections functions that can be defined in any cartesian closed category (Kock, 1972; Mac Lane,

1971). The finite form of bar recursion turns out to be a well-known morphism available in any strong monad, specialised to the selection monad.

1.1. *Organisation*

Section 2. Generalised quantifiers, finite products of quantifiers, and sequential games. *Section 3.* Selection functions, finite products of selection functions, calculation of optimal strategies. *Section 4.* Applications. *Section 5.* Infinite products of selection functions and quantifiers, bar recursion. *Section 6.* The continuation and selection monads. *Section 7.* Further work.

1.2. *Background and pre-requisites*

This paper has been deliberately written so that readers who are not familiar with certain categorical notions should be able to follow Sections 2–4 without the need of familiarising themselves with such concepts. These sections are officially developed in the generality of cartesian closed categories, but linguistically developed as if we were working with sets and functions as in ordinary mathematics (see below). Sections 5 and 6, on the other hand, rely on and apply to cartesian closed categories other than that of sets.

Recall that a category is called cartesian closed if it has finite products 1 and $X \times Y$, and function spaces $(X \to Y)$, often written Y^X in the literature, characterised by a natural bijection between maps $A \times X \to Y$ and $A \to (X \to Y)$ (see (Mac Lane, 1971)), given by currying and uncurrying in lambda-calculus terminology. Recall also that cartesian closedness is precisely what is needed in a category so that one can interpret the simply-typed lambda-calculus (Lambek and Scott, 1986). In the category of sets, the function space $(X \to Y)$ is the set of all functions $X \to Y$, and in certain cartesian closed of topological spaces, $(X \to Y)$ is the set of continuous maps with a suitable topology (see e.g. (Escardó et al., 2004)).

The main cartesian closed categories of interest for this work include (i) that of sets and functions, and more generally toposes (Johnstone, 2002), (ii) Howard/Bezem majorizable functionals (Bezem, 1985), (iii) spaces endowed with extended admissible representations in the sense of (Schröder, 2002), (iv) several categories of continuous maps of topological spaces (Escardó et al., 2004), such as k-spaces, QCB spaces (Battenfeld et al., 2007), Kleene–Kreisel spaces and continuous functionals (Normann, 1980), various categories of domains under the Scott topology (Abramsky and Jung, 1994), and (v) several categories of effective maps of effectively presented objects, such as Kleene–Kreisel computable maps (Normann, 1980), effectively given domains (Smyth, 1977), and the effective topos and realizability toposes, among others.

When working with our underlying cartesian closed category, we reason with generalised elements and the λ-calculus. So, for example, for any given $m \colon X \times X \to X$, the equation $m(x, y) = m(y, x)$ amounts to the element-free equation $m = m \circ \langle \pi_1, \pi_0 \rangle$, where π_0, π_1 are the projections. If m is regarded as a variable rather than a constant, this equation is to be understood as $\lambda m.m = \lambda m.m \circ \langle \pi_0, \pi_1 \rangle$. A global element of X is a map $1 \to X$, and a generalised element of X is a map $S \to X$, where S is called the stage

of definition of x. We write $x \colon X$, and occasionally $x \in X$ by an abuse of language, to mean that x is a generalised element of X at an unspecified stage S, which never needs to be mentioned explicitly due to the fact that we are working with the lambda-calculus. When the underlying category is well pointed, e.g. the category of sets and categories of continuous maps of spaces or domains, working with generalised elements is equivalent to working with actual elements (or global elements), and most of our examples will fall in this kind of category.

2. Generalised Quantifiers

The main notion discussed in this section is that of a (*generalised*) *quantifier*. Assume a fixed cartesian closed category, with a fixed object R, and define

$$KX \ := \ (X \to R) \to R.$$

We think of R as an object of generalised truth values, of functions $X \to R$ as predicates, of R-valued functions of several variables as relations, and of the elements of KX as generalised quantification functions, which, by an abuse of language, we refer to as quantification functions or simply quantifiers. This construction is part of a well-known monad, which we will develop in Section 6.

Examples 2.0.1.

1   Our underlying category is that of sets. Then the standard universal and existential quantifiers $\forall_X, \exists_X$ are elements of KX with $R = \mathbb{B} = \{\text{true}, \text{false}\}$.
2   More generally, our underlying category is a topos and $R = \Omega$ is the object of truth values (subobject classifier). Then the standard universal and existential quantifiers $\forall_X, \exists_X$ are elements of KX. Recall that in the topos of sets, $\Omega = \{\text{false}, \text{true}\} = \{0, 1\}$. We assume classical logic for the topos of sets (principle of excluded middle and axiom of choice).
3   Continuing from above, define

$$\phi(p) := \forall x \in X \exists y \in Y \, p(x, y),$$

for $p \colon X \times Y \to R$. Then $\phi \in K(X \times Y)$.
4   Assume R is the real line $\mathbb{R}$ in a cartesian closed category of spaces and continuous functions (such as k-spaces, QCB spaces, etc.). Define

$$I(p) := \int_0^1 p,$$

for $p \colon [0, 1] \to \mathbb{R}$. Then $I \in K[0, 1]$.

5 Continuing from the previous example, define

$$\phi(p) := \sup_{x \in [0,1]} \int_0^1 p(x,y)dy,$$

for $p \colon [0,1]^2 \to \mathbb{R}$. Then $\phi \in K([0,1]^2)$.

2.1. *Finite products of quantifiers*

The above Examples 2.0.1(3) and 2.0.1(5) are instances of the following construction.

Definition 2.1.1. Given quantifiers $\phi \in KX$ and $\gamma \in KY$, define a new quantifier

$$\phi \otimes \gamma \in K(X \times Y)$$

by, for any $p \colon X \times Y \to R$,

$$(\phi \otimes \gamma)(p) := \phi(\lambda x.\gamma(\lambda y.p(x,y))).$$

Examples 2.1.2.

1 If R is the object of truth values in a topos, then $\forall_X \otimes \forall_Y = \forall_{X \times Y}$, as this amounts to

$$(\forall_X \otimes \forall_Y)(p) = \forall x \in X (\forall y \in Y (p(x,y))) = \forall z \in X \times Y (p(z)) = \forall_{X \times Y}(p).$$

2 Similarly, we have that $\exists_X \otimes \exists_Y = \exists_{X \times Y}$.

3 And also $(\forall_X \otimes \exists_Y)(p) = \forall x \in X \, \exists y \in Y \, p(x,y)$.

4 If R are the real numbers $\mathbb{R}$ in a cartesian closed category of spaces and continuous functions, by Fubini's rule, we have $\int_{[0,1]} \otimes \int_{[0,1]} = \int_{[0,1] \times [0,1]}$, as this amounts to

$$\int_{[0,1]} \left(\int_{[0,1]} p(x,y)dy \right) dx = \int_{[0,1] \times [0,1]} p(x,y)d(x,y).$$

5 Generalising the previous example, let ν_i be Borel regular measures on locally compact Hausdorff spaces X_i for $i = 0,1$, and define $\phi_i \in KX_i$ by $\phi_i(p) := \int p d\nu_i$. Then $\phi \in K(X_0 \times X_1)$ defined by

$$\phi(p) := \int p d(\nu_0 \times \nu_1)$$

satisfies $\phi = \phi_0 \otimes \phi_1$, where $\nu_0 \times \nu_1$ is the product measure.

We now consider the iteration of the binary product of quantifiers defined above. Let us write

$$\prod_{i=0}^{n-1} X_i := X_0 \times \cdots \times X_{n-1}$$

with the conventions that the operation $\times$ is right associative, that for $n = 0$ this is the one-point set $1 = \{()\}$ where $()$ is the empty sequence, and that for $n = 1$ this is X_0. Hence for $n > 1$ this is

$$X_0 \times \prod_{i=1}^{n-1} X_i.$$

Definition 2.1.3. Given quantifiers $\phi \in \prod_{i=0}^{n-1} K X_i$, we define $\bigotimes_{i=0}^{n-1} \phi_i \in K\left(\prod_{i=0}^{n-1} X_i\right)$ as

$$\bigotimes_{i=0}^{n-1} \phi_i := \phi_0 \otimes \cdots \otimes \phi_{n-1},$$

which, expanding the definition amounts to

$$\left(\bigotimes_{i=0}^{n-1} \phi_i\right)(p) := \phi_0(\lambda x_0.\phi_1(\lambda x_1.\cdots \phi_{n-1}(\lambda x_{n-1}.p(x_0, x_1, \ldots, x_{n-1}))\cdots)).$$

Alternatively, we can define this product inductively, since

$$\bigotimes_{i=0}^{n-1} \phi_i = \phi_0 \otimes \left(\bigotimes_{i=1}^{n-1} \phi_i\right).$$

In this case we have

$$\left(\bigotimes_{i=0}^{n-1} \phi_i\right)(p) = \phi_0 \left(\lambda x_0. \left(\bigotimes_{i=1}^{n-1} \phi_i\right)(\lambda(x_1, \ldots, x_{n-1}).p(x_0, x_1, \ldots, x_{n-1}))\right),$$

which, writing $p_{x_0}(x_1, \ldots, x_n) := p(x_0, x_1, \ldots, x_n)$, can be expressed concisely as

$$\left(\bigotimes_{i=0}^{n-1} \phi_i\right)(p) = \phi_0 \left(\lambda x_0. \left(\bigotimes_{i=1}^{n-1} \phi_i\right)(p_{x_0})\right).$$

That is, the value of the quantifier $\bigotimes_{i=0}^{n-1} \phi_i$ on a predicate p is given by the value of the quantifier ϕ_0 on the predicate $\lambda x_0. \left(\bigotimes_{i=1}^{n-1} \phi_i\right)(p_{x_0})$. For the base case we can take the unary case

$$\bigotimes_{i=n-1}^{n-1} \phi_i = \phi_{n-1}$$

or, alternatively, the nullary case

$$\left(\bigotimes_{i=n}^{n-1} \phi_i\right)(p) = p(),$$

if we instead adopt the convention that $\prod_{i=0}^{n-1} X_i = X_0 \times \cdots \times X_{n-1} \times 1$ with the operation $\times$ right associative (cf. Section 3.3 below).

The empty product of quantifiers lives in $K1$ and is both the universal quantifier $\forall_1$ and the existential quantifier $\exists_1$, given by $\lambda p.p()$. With our official convention for finite products, this is a neutral element for the binary product up to isomorphism, in the sense that $\forall_1 \otimes \phi \in K(1 \times X)$ and $\phi \times \forall_1 \in K(X \times 1)$ are isomorphic to $\phi \in KX$ via the isomorphisms $K(1 \times X) \cong KX \cong K(X \times 1)$.

2.2. *Quantifiers in sequential games*

We now show how generalised quantifiers and their iterated products can be used to conveniently express some general notions regarding finite sequential games. It should

be noted that we use the language of sequential games simply for the sake of intuition, but, as we shall see in Section 4, our notion of game is general enough to capture many specific constructions in several application areas that are not usually formulated in terms of games.

Example 2.2.1. Consider an alternating, two-person game that finishes after exactly n moves, with one of the players winning. The i-th move is an element of the set X_i and the game is defined by a predicate $p\colon \prod_{i=0}^{n-1} X_i \to R$, with $R = \Omega$, that tells whether the first player, Eloise playing against Abelard, wins a given play $\vec{x} = (x_0, \ldots, x_{n-1}) \in \prod_{i=0}^{n-1} X_i$. Then Eloise has a winning strategy for the game p if and only if

$$\exists x_0 \in X_0 \forall x_1 \in X_1 \ldots \exists x_{n-2} \in X_{n-2} \forall x_{n-1} \in X_{n-1}\, p(x_0, \ldots, x_{n-1})$$

holds (assuming n even for notational convenience). Let $\phi_i := \exists_{X_i}$ for i even and $\phi_i := \forall_{X_i}$ for i odd. The above sufficient and necessary condition on Eloise having a winning strategy can be concisely expressed as

$$\left(\bigotimes_{i=1}^{n} \phi_i \right)(p).$$

The following definition abstracts from this example in several ways. First we assume R to be an arbitrary fixed object. Also, we focus on the number of rounds of the game, ignoring the number of players and who plays in each round, and we take the quantifier to be applied in each round as part of the definition of the game. We still require, however, that the game has a fixed length n.

Definition 2.2.2. Let $(X_i)_{i=0}^{n-1}$ be an n-tuple of objects, let $p\colon \prod_{i=0}^{n-1} X_i \to R$ be a predicate and $\phi\colon \prod_{i=0}^{n-1} KX_i$ be an n-tuple of quantifiers.

1   We think of the triple $((X_i)_{i=0}^{n-1}, p, \phi)$ as a *game*, or, more precisely, as a finite sequential game with n rounds.

   (a) X_i is the set of possible *moves* at round i.

   (b) A *play* is a sequence $\vec{x}\colon \prod_{i=0}^{n-1} X_i$.

   (c) p is the *outcome function*, and $p(\vec{x})$ is the outcome of the play $\vec{x}$.

   (d) $\phi_i\colon (X_i \to R) \to R$ is the *outcome quantifier* for round i.

2   Given a partial play $\vec{a}\colon \prod_{i=0}^{k-1} X_i$ for $k \leq n$, define the *sub-game outcome function* $p_{\vec{a}}\colon \prod_{i=k}^{n-1} X_i \to R$ by

$$p_{\vec{a}}(x_k, \ldots, x_{n-1}) := p(a_0, \ldots, a_{k-1}, x_k, \ldots, x_{n-1}),$$

or, more concisely,

$$p_{\vec{a}}(\vec{x}) := p(\vec{a} * \vec{x}),$$

where $*$ denotes concatenation of finite sequences. A partial play $\vec{a}$ defines a sub-game

$$((X_i)_{i=k}^{n-1}, p_{\vec{a}}, (\phi_i)_{i=k}^{n-1}),$$

which is like the original game but starts at the position determined by the initial

moves $\vec{a}$. Notice that if $k = n$ then p is constant, and when $k = 0$ this is the same as the full game.

3 The *optimal outcome* of the game is $w := \left(\bigotimes_{i=0}^{n-1} \phi_i \right)(p)$.

Hence, for any $\vec{a}$: $\prod_{i=0}^{k-1} X_i$,

$$w_{\vec{a}} := \left(\bigotimes_{i=k}^{n-1} \phi_i \right)(p_{\vec{a}})$$

is the optimal outcome of the sub-game determined by $\vec{a}$, and of course $w = w_{()}$. Note that if $k = n$ then $w_{\vec{a}} = p(\vec{a})$, whereas, if $k < n$, then

$$w_{\vec{a}} = \phi_k \left(\lambda x_k. \left(\bigotimes_{i=k+1}^{n-1} \phi_i \right)(p_{\vec{a}*x_k}) \right) = \phi_k \left(\lambda x_k.w_{\vec{a}*x_k} \right).$$

Hence, the optimal outcome of round k is determined by the outcome quantifier for round k together with a mapping $\lambda x_k.w_{\vec{a}*x_k}$ computing of the optimal outcome at round $k + 1$ given what is played at round k.

4 An *optimal move* a_k at round k is a move that forces the optimal outcome at round $k + 1$ to be the same as the optimal outcome at round k, i.e. $w_{\vec{a}} = w_{\vec{a}*a_k}$.

5 A play $\vec{a} = a_0, \ldots, a_{n-1}$ is *optimal* if each a_k is an optimal move in the sub-game determined by $a_0, \ldots, a_{k-1}$. Hence a play $\vec{a}$ is optimal if and only if

$$w_{()} = w_{(a_0)} = w_{(a_0,a_1)} = \cdots = w_{(a_0,\ldots,a_{n-1})}.$$

6 A *strategy* is a family of functions,

$$\text{next}_k : \prod_{i=0}^{k-1} X_i \to X_k,$$

with $k < n$, computing which move should be played at each round k, i.e. when the game is at position $\vec{a} = (a_i)_{i=0}^{k-1}$ the move selected is $a_k = \text{next}_k(\vec{a})$.

7 A strategy is *optimal* if for every $k < n$, and every partial play $(a_i)_{i=0}^{k-1}$, the move $\text{next}_k(\vec{a})$ is optimal at round k, i.e.

$$w_{\vec{a}} = \phi_k \left(\lambda x_k.w_{\vec{a}*x_k} \right) = w_{\vec{a}*\text{next}(\vec{a})}.$$

Given an optimal strategy, the definition by course of values induction

$$a_0 := \text{next}_0(), \qquad a_{k+1} := \text{next}_{k+1}(a_0, \ldots, a_k)$$

gives an optimal play.

Note that optimal strategies do not exist in general, but they do if the outcome quantifiers have selection functions in the sense of Section 3 below. In fact, we will show that a suitably defined product of selection functions calculates optimal strategies.

Example 2.2.3. In Example 2.2.1 above, the optimal outcome w of the game tells which of Eloise and Abelard has a winning strategy. Suppose, however, we choose $R = \{-1, 0, 1\}$ instead, with the convention that -1 means that Abelard wins, 0 means that the game

is a draw, and 1 that Eloise wins. Replace the existential and universal quantifiers by the minimum and maximum value functionals $\min_X, \max_X \colon (X \to R) \to R$ as

$$\phi_i = \begin{cases} \max_{X_i} & \text{if } i \text{ is even,} \\ \min_{X_i} & \text{if } i \text{ is odd.} \end{cases}$$

This is because Eloise tries to maximise the outcome of the game while Abelard tries to minimise the same outcome. If the optimal outcome w is 1, then Eloise has a winning strategy, if $w = -1$ then Abelard has a winning strategy, and if $w = 0$ then both Eloise and Abelard have strategies for not losing. Any optimal strategy next_k gives the best moves for Eloise when k is even, and for Abelard when k is odd.

3. Selection Functions for Quantifiers

The main notion investigated in this section is that of a selection function for a quantifier. Before introducing the notion, we discuss several well-known examples that motivate the general definition.

The mean value theorem asserts that for any continuous $p \colon [0,1] \to \mathbb{R}$ there is $a \in [0,1]$ such that

$$\int p = p(a).$$

Similarly, the maximum value theorem says that any continuous $p \colon X \to \mathbb{R}$ defined on a non-empty compact Hausdorff space X attains its maximum value: there is $a \in X$ such that

$$\sup p = p(a).$$

And of course this holds for minimum values too: there is $a \in X$ such that

$$\inf p = p(a).$$

If R is the object of truth values of the topos of sets, then for any non-empty set X and any predicate $p \colon X \to R$ there is $a \in X$ such that

$$\forall p = p(a).$$

This is popularly known as the drinker paradox: in any pub X there is a person a such that everybody drinks if and only if a drinks, where $p(x)$ is interpreted as the fact that x drinks. A variation of the drinker paradox is that in any pub X there is a person a such that somebody drinks if and only if a drinks. That is, for any $p \colon X \to R$ there is $a \in X$ such that $p(x)$ holds for some x if and only if $p(a)$ holds:

$$\exists p = p(a).$$

All of these statements hold in classical logic, but generally fail in intuitionistic logic or a computational setting. But notice that: (1) the drinker paradox, in its two forms, holds constructively for non-empty finite sets X, when R is the set of booleans (decidable truth values). Moreover, in this case, there is a stronger statement: (2) there is a function $\varepsilon \colon (X \to R) \to X$ that constructs, from p, the point a at which p attains its ϕ-value, in

the sense that

$$a = \varepsilon(p)$$

solves the equation. Of course, in the category of sets, if the desired a can always be found for any p, then there is a function ε as above that finds it from p, by the axiom of choice (in fact, this amounts to the axiom of choice).

Definition 3.0.4. Given a quantifier $\phi\colon (X \to R) \to R$, any function $\varepsilon\colon (X \to R) \to X$ such that

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

for all $p\colon X \to R$, is called a *selection function for ϕ*. A quantifier that has a selection function is called *attainable*.

We refer to $\phi(p)$ as the ϕ-value of p, and say that p attains its ϕ-value at a if $\phi(p) = p(a)$. With this terminology, ε is a selection function for the quantifier ϕ if and only if every p attains its ϕ-value at $\varepsilon(p)$. For our purposes, ε will play the role of providing an algorithm for computing $\phi(p)$ as $p(\varepsilon(p))$. Notice that if the quantifier $\phi\colon (X \to R) \to R$ is attainable, then the set X is non-empty, and $\phi(\lambda x.r) = r$ for any $r \in R$, because $(\lambda x.r)(\varepsilon(\lambda x.r)) = r$ for any choice of ε.

In the context of games, if X is a set of moves for a particular round, then a selection function $\varepsilon\colon (X \to R) \to X$ can be thought of as a *policy function*, i.e. a function that chooses a particular move $x \in X$ given that the effect of each move on the outcome of the whole game is known (i.e. $X \to R$). For instance, if the policy of the player is to maximise its payoff, then ε would be the functional computing the point $\varepsilon(p)$ where p attains its maximum value.

Remark 3.0.5. The paper (Escardó, 2008) defined selection functions for subsets S of X with R the discrete booleans (two-point space) in a cartesian closed category of continuous functions. Using the language of the above definition, we can formulate this as follows: a selection function for the set S is a selection function for the bounded existential quantifier $\exists_S\colon (X \to R) \to R$.

We will see in Section 6 that, like $KX = ((X \to R) \to R)$ defined above, J defined below gives rise to a monad, and this fact will play an illuminating role in our investigation of quantifiers that have selection functions. Before knowing that J and K are monads, the following defines a map that will turn out to be a monad morphism.

Definition 3.0.6. For R fixed as above, write $JX := ((X \to R) \to X)$. For any $\varepsilon \in JX$ we define a quantifier $\overline{\varepsilon} \in KX$ by

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

Thus, every $\varepsilon \in JX$ is a selection function of some quantifier, and hence we refer to the elements of JX as *selection functions*.

For selection functions of existential quantifiers, this construction occurs in the paper (Escardó, 2008), in particular in the proof of (Escardó, 2008, Lemma 3.4).

3.1. *Finite products of selection functions*

We now show that attainable quantifiers are closed under finite products. We then develop technical tools to be used in the applications described in Section 4. In order to establish preservation of attainability, we define a product of selection functions, which we show to correspond to the product of its associated quantifiers (cf. Definition 2.1.1).

Definition 3.1.1. Given selection functions $\varepsilon \in JX$ and $\delta \in JY$ define a new selection function

$$\varepsilon \otimes \delta \in J(X \times Y)$$

by

$$(\varepsilon \otimes \delta)(p) := (a, b(a))$$

where

$$
\begin{aligned}
b(x) &:= \delta(\lambda y.p(x, y)), \\
a &:= \varepsilon(\lambda x.p(x, b(x))).
\end{aligned}
$$

That is, from the relation $p\colon X \times Y \to R$, we get the function $b\colon X \to Y$ by choosing some y for a given x using the selection function δ. In a finite game of length two, this function gives a strategy for the second player. We can measure the success of the strategy for any move $x \in X$ by evaluating $p(x, b(x))$. It follows from the definition of $\bar{\delta}$ that $\bar{\delta}(\lambda y.p(x, y)) = p(x, b(x))$. This says that for any $x \in X$, the predicate $\lambda y.p(x, y)$ attains its $\bar{\delta}$-value at $b(x)$. Now, a as defined above is such that $\bar{\varepsilon}(\lambda x.p(x, b(x))) = p(a, b(a))$. Again, this says that the predicate $\lambda x.p(x, b(x))$ attains its $\bar{\varepsilon}$-value at a. Putting this all together we have:

Lemma 3.1.2. $\overline{\varepsilon \otimes \delta} = \bar{\varepsilon} \otimes \bar{\delta}$.

Proof. We calculate

$$(\overline{\varepsilon \otimes \delta})(p) = p(a, b(a)) = \bar{\varepsilon}(\lambda x.p(x, b(x))) = \bar{\varepsilon}(\lambda x.\bar{\delta}(\lambda y.p(x, y))) = (\bar{\varepsilon} \otimes \bar{\delta})(p),$$

by simply unfolding the definitions. $\qquad\square$

Remark 3.1.3. The above definition is equivalent to

$$(\varepsilon \otimes \delta)(p) := (a, \delta(\lambda y.p(a, y))),$$

where $a := \varepsilon(\lambda x.\bar{\delta}(\lambda y.p(x, y)))$, which was the construction in (Escardó, 2008, Proposition 4.4) to show that a finite product of searchable sets is searchable.

Example 3.1.4. Recall the drinker paradoxes for the quantifiers $\forall$ and $\exists$, defined above. Combining the two forms of the paradox with the product operator for selection functions we get: In any group of people, there are a man a and a woman b such that every man loves some woman if and only if a loves b. More precisely, for any two non-empty sets X and Y and any predicate $p\colon X \times Y \to \Omega$, there is $(a, b) \in X \times Y$ such that

$$(\forall x \in X \exists y \in Y\, p(x, y)) = p(a, b).$$

In fact, by the two versions of the drinker paradox, the universal and existential quantifiers $\forall_X$ and $\exists_Y$ have selection functions A_X and E_Y respectively, and hence $A_X \otimes E_Y$ is a selection function for the quantifier $\forall_X \otimes \exists_Y$, and thus we can take $(a, b) = (A_X \otimes E_Y)(p)$.

Notice that $J1$ has precisely one element, which is a neutral element for the product up to isomorphism. We adopt the notation $\bigotimes_{i=0}^{n-1} \varepsilon_i$ for the iterated product of selection functions, as we did for quantifiers. By Lemma 3.1.2 and straightforward induction:

Theorem 3.1.5. For any sequence $\varepsilon \in \prod_{i=0}^{n-1} JX_i$ of selection functions,

$$\overline{\bigotimes_{i=0}^{n-1} \varepsilon_i} = \bigotimes_{i=0}^{n-1} \overline{\varepsilon_i}.$$

The following expresses this in terms of attained values, which is useful for the formulation and justification of the applications we have in mind:

Corollary 3.1.6. If ε_i is a selection function for a quantifier ϕ_i, and if we define

$$E = \bigotimes_{i=0}^{n-1} \varepsilon_i, \qquad \Phi = \bigotimes_{i=0}^{n-1} \phi_i,$$

then every $p\colon \prod_{i=0}^{n-1} X_i \to R$ attains its Φ-value at $\vec{a} = E(p)$, in the sense that

$$\Phi(p) = p(\vec{a}).$$

Example 3.1.7. We continue from Example 2.2.1 on two-person games. Let $A_i, E_i \in JX_i$ be selection functions for the quantifiers $\forall_{X_i}$ and $\exists_{X_i}$ respectively, and define

$$\varepsilon_i = \begin{cases} E_i & \text{if } i \text{ is even,} \\ A_i & \text{if } i \text{ is odd,} \end{cases} \qquad \phi_i = \begin{cases} \exists_{X_i} & \text{if } i \text{ is even,} \\ \forall_{X_i} & \text{if } i \text{ is odd.} \end{cases}$$

By Corollary 3.1.6, for any game $p\colon \prod_{i=0}^{n-1} X_i \to \Omega$, the play $\vec{a} := (\bigotimes_{i=0}^{n-1} \varepsilon_i)(p)$ is such that Eloise has a winning strategy in the game p if and only if she wins the play $\vec{a}$, as this amounts to the equation

$$\left(\bigotimes_{i=0}^{n-1} \phi_i \right)(p) = p(\vec{a}).$$

Section 3.2 below shows, in particular, that $\vec{a}$ above is an optimal play, and that the product of selection functions can also be used to compute optimal strategies.

Remark 3.1.8. In several kinds of games, the set of allowed moves at round $i+1$ depends on the move played at round i. We can account for this with the following generalisation of the binary product:

1. Given a quantifier $\phi \in KX$ and family of quantifiers $\gamma\colon X \to KY$, define their dependent product $\phi \otimes \gamma \in K(X \times Y)$ as

$$(\phi \otimes \gamma)(p) := \phi(\lambda x.\gamma(x)(\lambda y.p(x, y))),$$

for $p\colon X \times Y \to R$.

2 For example, the combination of quantifiers $\forall x \in A \,\exists y \in B(x)\, p(x, y)$ arises as a dependent product $\phi \otimes \gamma$, where $A \subseteq X$ and $B(x) \subseteq Y$ for each $x \in A$, and where $\phi = \forall_A$ and $\gamma(x) = \exists_{B(x)}$.

3 Similarly, given a selection function $\varepsilon \in JX$ and a family of selection functions $\delta \colon X \to JY$, we define their dependent product as

$$(\varepsilon \otimes \delta)(p) := (a, b(a))$$

for $p \colon X \times Y \to R$, where

$$
\begin{aligned}
b(x) &:= \delta(x)(\lambda y.p(x, y)), \\
a &:= \varepsilon(\lambda x.p(x, b(x))).
\end{aligned}
$$

4 Then Lemma 3.1.2 holds for this notion of dependent product, with a routine generalisation of its proof.

3.2. *Calculating optimal strategies*

Let $((X_i)_{i=0}^{n-1}, p, \phi)$ be a game in the sense of Definition 2.2.2, and suppose that each quantifier ϕ_i has a selection function ε_i. By definition of selection function and of optimal strategy:

Lemma 3.2.1. The construction

$$\mathrm{next}_k(\vec{x}) := \varepsilon_k(\lambda x_k.w_{\vec{x}*x_k}),$$

where $w_{\vec{x}}$ is defined in 2.2.2(3), gives an optimal strategy.

Recall that the optimal outcome $w_{\vec{x}}$ of a sub-game is defined in terms of products of quantifiers. Our next objective is to calculate this optimal strategy as a product of selection functions instead. In order to do this, we develop the following two recursive characterisations of finite products of selection functions, which are interesting on their own right:

Lemma 3.2.2.

$$\left(\bigotimes_{i=k}^{n-1} \varepsilon_i \right)(p) = a_k * \left(\left(\bigotimes_{i=k+1}^{n-1} \varepsilon_i \right)(p_{a_k}) \right),$$

where

$$a_k = \varepsilon_k \left(\lambda x_k.p_{x_k}\left(\left(\bigotimes_{i=k+1}^{n-1} \varepsilon_i \right)(p_{x_k}) \right) \right),$$

Proof. This follows directly from Remark 3.1.3, taking $X = X_k$, $Y = \prod_{i=k+1}^{n-1} X_i$, $\varepsilon = \varepsilon_k$ and $\delta = \bigotimes_{i=k+1}^{n-1} \varepsilon_i$, $a = a_k$. $\qquad\square$

Lemma 3.2.3.

$$\left(\bigotimes_{i=0}^{n-1} \varepsilon_i \right)(p) = \vec{a}$$

where $\vec{a}$ is given by course of values recursion as

$$
\begin{aligned}
a_k & = \varepsilon_k\left(\lambda x_k.p_{a_0,\ldots,a_{k-1},x_k}\left(\left(\bigotimes_{i=k+1}^{n-1}\varepsilon_i\right)(p_{a_0,\ldots,a_{k-1},x_k})\right)\right) \\
& = \varepsilon_k(\lambda x_k.w_{a_0,\ldots,a_{k-1},x_k}).
\end{aligned}
$$

Proof. The first equation follows by Lemma 3.2.2 and course of values induction. By the assumption that ε_i is a selection function for the quantifier ϕ_i and Theorem 3.1.5, the optimal outcome of the game that starts at position $\vec{x} \in \prod_{i=0}^{k-1} X_i$ can be calculated as

$$
w_{\vec{x}} = p_{\vec{x}}\left(\left(\bigotimes_{i=k}^{n-1}\varepsilon_i\right)(p_{\vec{x}})\right),
$$

which gives the second equation. □

By Lemmas 3.2.2 and 3.2.3 we get:

Theorem 3.2.4. The optimal-strategy functions next_k constructed in Lemma 3.2.1 can be calculated as

$$
\text{next}_k(\vec{x}) := \left(\left(\bigotimes_{i=k}^{n-1}\varepsilon_i\right)(p_{\vec{x}})\right)_0.
$$

Moreover,

1 The whole sequence

$$
\vec{a} = \left(\bigotimes_{i=k}^{n-1}\varepsilon_i\right)(p_{\vec{x}})
$$

is an optimal play for the game that starts at position $\vec{x}$.

2 The predicates $p_k \colon X_k \to R$ defined by

$$
p_k(x_k) = w_{a_0,\ldots,a_{k-i},x_k} = p_{a_0,\ldots,a_{k-1},x_k}\left(\left(\bigotimes_{i=k+1}^{n-1}\varepsilon_i\right)(p_{a_0,\ldots,a_{k-1},x_k})\right)
$$

satisfy

$$
\varepsilon_k(p_k) = a_k, \qquad p_k(a_k) = p_j(a_j).
$$

for all $k, j < n$.

Theorem 3.2.4(2) says that the optimal move a_k can be computed from the selection function ε_k and the mapping $\lambda x_k.w_{a_0,\ldots,a_{k-1},x_k}$ of possible moves at round k to optimal outcomes at round $k + 1$.

3.3. *Implementation of the finite product*

The computation of finite products of selection functions can be easily implemented in higher-type functional programming languages such as Haskell (Hutton, 2007) in the case all types X_i are the same and equal to X.

```
type J r x = (x -> r) -> x

otimes :: J r x -> J r [x] -> J r [x]
(epsilon 'otimes' delta) p = a : b(a)
    where b(x) = delta(\xs -> p(x : xs))
          a = epsilon(\x -> p(x : b(x)))

bigotimes :: [J r x] -> J r [x]
bigotimes [] = \p -> []
bigotimes (epsilon : epsilons) =
    epsilon 'otimes' (bigotimes epsilons)
```

We are using lower case letters `r` and `x` for R and X due to Haskell's syntactical requirements. In Haskell, a finite list of length n is written

$$[x_0, x_1, \ldots, x_{n-1}] = x_0 : x_1 : \cdots : x_{n-1} : [],$$

where [] is the empty list. The operator `otimes` computes the binary product of a selection function $\varepsilon \colon JX_0$ with a selection function $\delta \colon J\left(\bigotimes_{i=1}^{n-1} X_i\right)$, obtaining a selection function in $J\left(\bigotimes_{i=0}^{n-1} X_i\right)$, and the function `bigotimes` iterates this finitely often. List types in Haskell actually include infinite lists, and we will see in Section 5 that this algorithm in fact works for infinite lists of selection functions as well (and corresponds to a form of bar recursion). Dependently typed languages such as Agda (Bove and Dybjer, 2008) allow the types X_i to be distinct, with a similar recursive definition.

4. Applications

In this section we show that finite products of selection functions appear in many guises in different areas, such as game theory, fixed point theory, proof theory and algorithms.

4.1. *Game theory*

Consider a sequential game with n players (say $0, 1, \ldots, n-1$) and n rounds, with player i picking his move at round i from a fixed set X_i. In standard game theory, a play $(x_0, \ldots, x_{n-1}) \in \Pi_{i=0}^{n-1} X_i$ (cf. Definition 2.2.2) is also known as a *strategy profile*, and outcome functions $p \colon \Pi_{i=0}^{n-1} X_i \to \mathbb{R}^n$ are called *payoff functions*, since $p(x_0, \ldots, x_{n-1}) = (v_0, \ldots, v_{n-1})$ gives the payoff each player gets at the end of all rounds. Each player is trying to maximise their payoff, so the outcome selection functions $\varepsilon_i \colon (X_i \to \mathbb{R}^n) \to X_i$ are

$$\varepsilon_i(q) := x \in X_i \text{ such that } (qx)_i \geq (\max\{(qx)_i \ : \ x \in X_i\})$$

where $q : X_i \to \mathbb{R}^n$. Finally, an optimal play is a strategy profile where each player has maximised their possible payoff, relative to the choice of the other players.

Theorem 4.1.1. The optimal play

$$\vec{x} := \left(\bigotimes_{i=0}^{n-1} \varepsilon_i\right)(p)$$

is a strategy profile in Nash equilibrium.

By the definition of Nash equilibrium, it is enough to note that the optimal strategy function $\mathrm{next}_k(x_0, \ldots, x_{k-1})$ computes the move for player k (in the sub-game $p_{x_0,\ldots,x_{k-1}}$) maximising his payoff, given that all the following players are playing optimally. Hence, once an optimal play has been obtained, any change of move from either player individually cannot result in a better payoff for that player.

The above construction can be viewed as a formal description of *backward induction*, a technique used in Game Theory (Nisan et al., 2007) to compute Nash equilibria in sequential games. Intuitively, backward induction is explained as follows: An equilibrium strategy profile is computed by inductively pruning branches of the game tree. Starting from the last player, we pick in each sub-tree only the branch which would be selected by the last player if that sub-game is reached. The same is then done for each player in turn, in reverse order. We end up with just one branch left which by construction is an optimal play.

4.2. *Fixed point theory*

A map $\mathrm{fix}\colon (R \to R) \to R$ is a called a *fixed point operator* if $\mathrm{fix}(p)$ is a fixed point of p for every $p\colon R \to R$, i.e.,

$$\mathrm{fix}(p) = p(\mathrm{fix}(p)).$$

For non-trivial fixed point operators to exist, we must work in a cartesian closed category other than that of classical sets, as for every set, except the one-point set, there is an endofunction with no fixed point. Well known examples are various categories of domains.

Now, $JR = KR = ((R \to R) \to R)$, and hence a fixed point operator can be considered both as a selection function and as a quantifier. Moreover, $f\colon (R \to R) \to R$ is a fixed point operator if and only if it is its own selection function, as this amounts to

$$f(p) = \overline{f}(p) = p(f(p)).$$

Bekič's Lemma (Bekič, 1984) says that if X and Y have fixed point operators, then so does $X \times Y$, and moreover explicitly constructs a fixed point operator for the product from given fixed point operators for the factors. We now show that Bekič's construction arises as a product of suitable selection functions:

Lemma 4.2.1. If X and Y have fixed point operators fix_X and fix_Y, then $X \times Y$ has a fixed point operator $\mathrm{fix}_{X \times Y} \in J(X \times Y)$, with $R = X \times Y$, given by

$$\mathrm{fix}_{X \times Y} := \varepsilon_X \otimes \delta_Y,$$

where we define the selection functions $\varepsilon \in JX$ and $\delta \in JY$ (also with $R = X \times Y$) by

$$\varepsilon_X(p) := \mathrm{fix}_X(\pi_X \circ p), \qquad \delta_Y(q) := \mathrm{fix}_Y(\pi_Y \circ q).$$

Here $\pi_X \colon X \times Y \to X$ and $\pi_Y \colon X \times Y \to Y$ are the projections.

The selection functions ε and δ are not fixed point operators themselves, nor are the derived quantifiers $\phi = \overline{\varepsilon} \in KX$ and $\gamma = \overline{\delta} \in KY$. But, by Theorem 3.1.5 that gives $\overline{\varepsilon \otimes \delta} = \phi \otimes \gamma$, and by the facts that $J(X \times Y) = K(X \times Y)$ for $R = X \times Y$, and that $\mathrm{fix}_{X \times Y}$ is a fixed point operator if and only if it is its own selection function, we conclude from Lemma 4.2.1 that $\mathrm{fix}_{X \times Y}$ is also given as a product of quantifiers:

$$\mathrm{fix}_{X \times Y} = \phi \otimes \gamma.$$

For the *proof* of Lemma 4.2.1, however, again in view of Theorem 3.1.5, it is enough to conclude that

$$\varepsilon \otimes \delta = \phi \otimes \gamma,$$

where of course the left product is of selection functions and the right one is of quantifiers, because then $\mathrm{fix}_{X \times Y}$ is its own selection function and hence is a fixed point operator. Indeed, when applied to a function $r = (s, t) \colon X \times Y \to X \times Y$, both sides of the equation reduce to the same term, namely (a, b) with

$$
\begin{aligned}
a &= \mathrm{fix}_X(\lambda x . s(x, \mathrm{fix}_Y(\lambda y . t(x, y)))), \\
b &= \mathrm{fix}_Y(\lambda y . t(a, y)).
\end{aligned}
$$

This is Bekič's formula for calculating a fixed point (a, b) of the function r. Of course, here we are using the fact that any $r \colon X \times Y \to X \times Y$ is of the form (s, t) with $s \colon X \times Y \to X$ and $t \colon X \times Y \to Y$,

$$r(x, y) = (s(x, y), t(x, y)),$$

by considering $s = \pi_X \circ r$ and $t = \pi_Y \circ r$.

Notice that there is an asymmetry in the definitions of a and b. If we switch the roles of a and b (and of s and t), another fixed-point operator is obtained. We have not investigated the relationship between these two fixed-point operators, but we suspect they do not coincide in general. As is well known in domain theory (and first observed by Bekič), however, if X and Y are objects of a category of domains and continuous functions, and fix_X and fix_Y are the least fixed-point operators, then either construction produces the least fixed-point operator of the product domain $X \times Y$, and hence the two constructions coincide.

By Lemma 4.2.1 and induction:

Theorem 4.2.2. If X_i for $0 \le i < n$ has a fixed point operator fix_i, then $\prod_{i=0}^{n-1} X_i$ has a fixed point operator $\mathrm{fix} \in J(\prod_{i=0}^{n-1} X_i)$, with $R = \prod_{i=0}^{n-1} X_i$, given as a product of selection functions,

$$\mathrm{fix} = \bigotimes_{i=0}^{n-1} (\lambda p_i . \mathrm{fix}_i(\pi_i \circ p_i)),$$

where π_i is the projection of the product into X_i.

4.3. *Proof theory*

Uses of the product of selection functions in proof theory will be further discussed in Section 5.7, where we explain how this construction is related to the so-called bar recursion. In this section we look at a simple example, where the computational interpretation of a non-computational principle can be explained again in terms of products of selection functions. The principle we consider is the *infinite pigeon-hole principle*, which says that for any finite set $n = \{0, 1, \dots, n-1\}$ of colours and any colouring of the natural numbers, some colour occurs infinitely often:

$$\forall n \colon \mathbb{N} \, \forall f \colon \mathbb{N} \to n \, \exists k \in n \, \forall i \, \exists j \geq i (fj = k).$$

This is non-computational, in the sense that, given n and f, we cannot effectively produce the colour k which is used infinitely often. We look, therefore, at the dialectica interpretation (Avigad and Feferman, 1998) of its negative translation, i.e.

$$\forall n \colon \mathbb{N} \, \forall f \colon \mathbb{N} \to n (\neg\neg \exists k \in n \, \forall i \, \exists j \geq i (fj = k)).$$

The dialectica interpretation of this is

$$\forall n \colon \mathbb{N} \, \forall f \colon \mathbb{N} \to n \, \forall \varepsilon \colon n \to (\mathbb{N}^{\mathbb{N}} \to \mathbb{N}) \, \exists k \in n \, \exists p \colon \mathbb{N} \to \mathbb{N}$$
$$p(\varepsilon_k(p)) \geq \varepsilon_k(p) \wedge f(p(\varepsilon_k(p))) = k,$$

i.e. given n, f and a sequence ε_i, we must find k and p such that

$$p(\varepsilon_k(p)) \geq \varepsilon_k(p) \wedge f(p(\varepsilon_k(p))) = k.$$

Intuitively, the function p is trying to compute a value j which makes the statement true (i.e., selection function for existential quantifier in $\exists j \geq i (fj = k)$), whereas the functional ε_k tries to produce a counter-example i given any such p and fixed colour k (i.e., selection function for universal quantifier in $\forall i (pi \geq i \wedge f(pi) = k)$). The above constructive version of the infinite pigeon-hole principle says that given a partition f of the natural numbers into n sets, and given n (counter-example) selection functions $\varepsilon_0, \dots, \varepsilon_{n-1}$, one for each colour, we can always find $k < n$ and p_k, such that ε_k is not successful in finding a counter-example, i.e. $p_k(\varepsilon_k(p_k)) = k$.

But, again, we can also view the above as an instance of our general notion of sequential game (Definition 2.2.2). Consider the game where $X_i = R = \mathbb{N}$, and $\max \colon \mathbb{N}^n \to \mathbb{N}$ is the outcome function, and $\phi_k(p) = p(\varepsilon_k p)$ is the outcome quantifier for round k. We show how the above computational interpretation of the infinite pigeonhole principle can be realized from the optimal play in this sequential game. Note that although this is a finite game, with n rounds, we have at each round k an infinite number of possible moves as $X_k = \mathbb{N}$.

Theorem 4.3.1. Let n, f and ε_i be given. Define

$$\vec{x} := \left(\bigotimes_{i=0}^{n-1} \varepsilon_i \right) (\max).$$

Then, for all $k < n$ we have $p_k(x_k) \geq x_k$, and for some $k < n$ we have

$$f(p_k(\varepsilon_k(p_k))) = k,$$

where

$$p_k(y) := \max_{x_0,\ldots,x_{k-1},y} \left(\left(\bigotimes_{i=k+1}^{n-1} \varepsilon_i \right) \left(\max_{x_0,\ldots,x_{k-1},y} \right) \right).$$

In fact, by Theorem 3.2.4, $x_k = \varepsilon_k(p_k)$, for all $k < n$. Hence

$$p_k(x_k) = \max\{x_0, \ldots, x_{n-1}\},$$

and $p_k(x_k) \geq x_k$, for all $k < n$. It remains to show that $f(p_k(\varepsilon_k(p_k))) = k$ for some k, but this follows from the fact that $p_k(\varepsilon_k(p_k)) = p_k(x_k)$ is the same for all $k < n$, again by Theorem 3.2.4.

Remark 4.3.2. A similar calculation was performed in (Oliva, 2006), but using a finite version of Spector's bar recursion (cf. Section 5.7) instead of finite products of selection functions.

4.4. *Algorithms*

Products of selection functions also correspond to the algorithmic technique of *back-tracking*. For instance, if each $\varepsilon_k : (\mathbb{B} \to \mathbb{B}) \to \mathbb{B}$ is a selection function for the boolean existential quantifier, and $p(x_0, \ldots, x_{n-1})$ is a decidable predicate on n-boolean variables, then

$$\left(\bigotimes_{i=0}^{n-1} \varepsilon_i \right) (p)$$

computes an assignment which makes p true, if p is satisfiable.

The same construction can also be used to compute a shortest path between two nodes in a given weighted directed graph, where in this case the quantifiers are the minimum functionals. Let X be a finite set of vertices, and $d \colon X \times X \to R$ be the weighted incidence matrix of the directed graph, with $d(x, x) = 0$, where $R = [0, \infty]$. If $d(x, y) = \infty$ this means that there is no edge from node x to node y; otherwise this gives the weight of the edge from x to y. Let n be the cardinality of X, and let $X_i = X$ for $i < n$. Define the quantifiers $\phi_i : (X_i \to R) \to R$ as

$$\phi_i(p) := \min p = \min\{p(x) : x \in X_i\},$$

and let ε_i be a selection function for ϕ_i. So we have a constant sequence of quantifiers, and of selection functions. The length of a path $x_0, \ldots, x_{k-1}$ is defined as $d(x_0, x_1) + \ldots + d(x_{k-2}, x_{k-1})$. If this is different from ∞, and if $x_i \neq x_j$ for $i \neq j$, we call this a *proper path*. Given vertices u and v, define $q : X^n \to R$ by

$q(x_0, \ldots, x_{n-1}) :=$ if there is $k < n$ such that $u, x_0, \ldots, x_k, v$ is a proper path then the length of such path for the smallest k else ∞.

Theorem 4.4.1. A shortest path, or the non-existence of a path from u to v, can be read off from

$$\vec{a} := \left(\bigotimes_{i=0}^{n-1} \varepsilon_i \right) (q).$$

More precisely, if $q(\vec{a}) = \infty$ then v is not reachable from u; otherwise look for the smallest $k < n$ such that $d(a_k, v) \neq \infty$, and the shortest path from u to v is $u, a_0, \ldots, a_k, v$.

In fact, by simultaneous induction on $n - k - 1$,

$$\left(\bigotimes_{i=k}^{n-1} \varepsilon_i \right) (q_{a_0,\ldots,a_{k-1}})$$

calculates the shortest way to link the path $u, a_0, \ldots, a_{k-1}$ to the node v, and

$$q_{a_0,\ldots,a_{k-1},a_k} \left(\bigotimes_{i=k+1}^{n-1} \varepsilon_i \right) (q_{a_0,\ldots,a_{k-1},a_k})$$

calculates the length of any such shortest way.

Note that this solution corresponds to computing a shortest path via backtracking with pruning, which is less efficient than Dijkstra's algorithm. The tree over which backtracking is performed is based on the order in which the predicate q queries its arguments. Also, the pruning takes place whenever q finds that the argument $x_0, x_1, \ldots, x_k$ is not a proper path by only looking at a few positions, and hence speeding up the backtracking (cf. (Escardó, 2007)). In fact, the product of selection functions behaves as such in general, including all applications mentioned above.

Note also that, alternatively, we could use the dependent version of the product of selection functions (Remark 3.1.8) to ensure that the next element added to the path is connected to the previous one, and has not been visited before, meaning that only proper paths are considered.

5. Infinite Products of Selection Functions

In (Escardó, 2008, Definition 4.5), a functional

$$\Pi \colon ((D \to \mathcal{B}) \to D)^\omega \to ((D^\omega \to \mathcal{B}) \to D^\omega)$$

is constructed, where D is a domain and $\mathcal{B}$ is the lifted domain of booleans. Using our notation and choosing $R = \mathcal{B}$, the type definition of this functional can be written as

$$\Pi \colon (JD)^\omega \to JD^\omega.$$

It is proved in (Escardó, 2008, Theorem 4.6) that, given $\varepsilon \in (JD)^\omega$ such that $\varepsilon_i \in JD$ is a selection function for an existential quantifier $\exists_{S_i}$, with $S_i \subseteq D$, it holds that $\Pi(\varepsilon) \in JD^\omega$ is a selection function for the existential quantifier $\exists_{\Pi_i S_i}$ of the set $\prod_i S_i \subseteq D^\omega$.

5.1. *Generalisation of the product functional*

We rework the product functional Π in a number of ways. Firstly, we work with an infinite sequence X_i of spaces rather than a single domain D, and we replace the countable product $(JD)^\omega$ by the dependent product $\prod_i JX_i$. To be consistent with our notation, we rename the functional to $\bigotimes$, and give it the type

$$\bigotimes \colon \prod_i JX_i \to J\left(\prod_i X_i\right).$$

Secondly, we allow R to be any discrete space, not just the booleans. In the infinite case, the assumption that R be discrete is essential (Remark 5.5.8). Thirdly, we show that, more generally, given $\varepsilon \in \prod_i JX_i$ such that $\varepsilon_i \in JX_i$ is a selection function for a quantifier $\phi_i \in KX_i$, it holds that $\bigotimes_i \varepsilon_i \in J\left(\prod_i X_i\right)$ is a selection function for a suitably defined quantifier $\bigotimes_i \phi_i \in K\left(\prod_i X_i\right)$. Lastly, we observe that the recursive definition of the Π functional given in (Escardó, 2008, Section 8.1, page 30), here written $\bigotimes$ as explained above, can be written as

$$\bigotimes_i \varepsilon_i = \varepsilon_0 \otimes \bigotimes_i \varepsilon_{i+1},$$

and hence the infinite version can be seen as simply the iteration of the binary version of the product of selection functions. We also show that the analogous equation

$$\bigotimes_i \phi_i = \phi_0 \otimes \bigotimes_i \phi_{i+1}$$

holds for attainable quantifiers, but unfortunately does not characterise infinite products of quantifiers in general.

We will see in Section 5.5 that these equations for infinite products can be understood as definitions by bar recursion, introduced in Section 5.4. We first discuss the spaces to which the development discussed above applies (Section 5.2), and observe that infinite sequential games, in the continuous case, amount to finite games of unbounded length (Section 5.3).

5.2. *A convenient category of spaces and domains*

In order to be able to form the required function spaces for the product functional, we work in a cartesian closed category of continuous maps of topological spaces closed under countable products. The largest such category for we which are able to prove our main results is that of continuous maps of QCB spaces (Battenfeld et al., 2006; Battenfeld et al., 2007). Such spaces are precisely the T_0 topological quotients of countably based spaces, and can be characterised in a number of ways, such as, among others: (i) the sequential T_0 spaces with countable pseudo-bases, or (ii) the sequential T_0 spaces with admissible quotient representations, (iii) a certain full subcategory of the category $\mathrm{PER}(\omega\mathrm{AlgLat})$, whose objects are countably based algebraic lattices with a partial equivalence relation, and whose morphisms are Scott continuous maps that preserve the equivalence relation (Bauer, 2002). QCB spaces admit a theory of computability, and,

as shown by (Escardó et al., 2004), have some well-known cartesian closed subcategories closed under countable products, such as (i) Kleene–Kreisel continuous functionals, and (ii) Ershov–Scott continuous functionals. Hence, in particular, they simultaneously account for total and partial computation. For some lemmas, we allow k-spaces (also known as compactly generated spaces), which contain QCB as a full subcategory closed under finite and countable products and function spaces (Escardó et al., 2004), as the restriction to QCB spaces would be artificial and serve no purpose.

5.3. *Finite games of unbounded length*

In a topological setting, the move from finite to countable products corresponds to the move from finite games of fixed length to finite games of unbounded length. In order to see this, notice that if a discrete-valued function $p\colon \prod_i X_i \to R$ is continuous, then for any sequence $\alpha \in \prod_i X_i$, the value $p(\alpha)$ depends only on a finite prefix of the sequence α. Formally, for $\alpha, \beta \in \prod_i X_i$ and $n \geq 0$, define

$$\alpha =_n \beta \quad \text{iff} \quad \alpha_i = \beta_i \text{ for all } i < n.$$

If p is continuous then for every $\alpha \in \prod_i X_i$ there is some n such that

$$p(\beta) = p(\alpha) \text{ for all } \beta =_n \alpha.$$

Denote by n_α the smallest such n. If $p\colon \prod_i X_i \to R$ is the outcome function of a game, then continuity of p implies that the outcome of every infinite play is determined by a finite prefix of the play. In this case, we may say that the play α terminates in n_α rounds. It is in this sense that, by considering continuous outcome functions, we move from finite games of fixed length to finite games of unbounded length.

5.4. *Bar induction*

A continuous discrete-valued continuous function $p\colon \prod_i X_{i+n} \to R$ can be regarded as a well-founded tree as follows. The root of the tree is the only node of level 0. Each node of level i is either a leaf labelled by an element of R, or else has one branch for each point of X_{i+n}, leading to a node of level $i + 1$. The well-foundedness condition says that each maximal path of the tree starting from the root is finite and hence eventually reaches a leaf. For each $\alpha \in \prod_i X_{i+n}$, the finite prefix of α of length n_α (defined in Section 5.3) gives a maximal path ending at a leaf labelled by the value $p(\alpha)$, and all maximal paths of the tree are of this form. Hence if p is constant then it is seen as a singleton tree consisting of just a leaf, and otherwise the subtree of p that follows the branch $x_n \in X_n$ is that corresponding to the predicate $p_{x_n}\colon \prod_i X_{i+n+1} \to R$. Intuitively, to evaluate $p(\alpha)$ for any given $\alpha \in \prod_i X_{i+n}$, one follows the branches $\alpha_0, \alpha_1, \ldots$ until a leaf is reached, whose label gives the value $p(\alpha)$. Notice that different trees can give rise to the same continuous function. The procedure described above builds the optimal tree, corresponding to the optimal modulus of continuity $\alpha \mapsto n_\alpha$ of the function p.

The following is a counter-part of induction on well-founded trees, and is well known in various guises and particular situations:

Lemma 5.4.1 (Bar induction). Let X_i be a sequence of k-spaces and R be discrete, and consider a sequence of sets

$$\mathcal{A}_n \subseteq P_n := \left(\prod_i X_{i+n} \to R \right),$$

where the product and exponential are calculated in the category of k-spaces. If for all n,

1 the constant functions are in $\mathcal{A}_n$, and
2 for all $p \in \mathcal{A}_n$, the condition $\forall x \in X_n \, (p_x \in \mathcal{A}_{n+1})$ implies $p \in \mathcal{A}_n$,

then $\mathcal{A}_n = P_n$ for all n.

Proof. Suppose that for some n, there is $p \in P_n$ such that $p \notin \mathcal{A}_n$. Then, by the assumption, there is some α_n such that $p_{\alpha_n} \notin \mathcal{A}_{n+1}$. Proceeding in the same manner, we get an infinite sequence $\alpha \in \prod_{i \geq n} X_i$ such that $p_{\alpha_n, \alpha_{n+1}, \ldots, \alpha_k} \notin \mathcal{A}_{n+k+1}$ for every k. But, by continuity, $p_{\alpha_n, \alpha_{n+1}, \ldots, \alpha_k}$ is constant for some k, and hence is in $\mathcal{A}_{n+k+1}$ by assumption, which is a contradiction. $\qquad\square$

We now consider definitions of continuous functionals $h_n \colon P_n \to Y_n$ by bar recursion (Normann, 1999, Section 6), where the spaces P_n are as in Lemma 5.4.1 and the spaces Y_n are arbitrary. Given $L_n \colon R \to Y_n$ and $B_n \colon P_n \times (X_n \to Y_{n+1}) \to Y_n$, we consider the equations

$$
\begin{aligned}
h_n(\lambda \alpha.r) &= L_n(r), \\
h_n(p) &= B_n(p, \lambda x.h_{n+1}(p_x)).
\end{aligned}
$$

The intuitive idea is that the base case L_n accounts for leaves and the recursion step B_n for branches. By bar induction, one easily sees that there is at most one such function h_n. Of course, one cannot continuously test whether a function is constant or not, and hence there is no guarantee that there is a continuous solution. Moreover, the second equation also applies to the case when p is the constant function $\lambda \alpha.r$, where we get, using both equations and the fact that $(\lambda \alpha.r)_x = \lambda \beta.r$,

$$L_n(r) = h_n(\lambda \alpha.r) = B_n(\lambda \alpha.r, \lambda x.h_{n+1}(\lambda \beta.r)) = B_n(\lambda \alpha.r, \lambda x.L_{n+1}(r)).$$

Now, by bar induction, one easily sees that:

Lemma 5.4.2. Let $B_n \colon P_n \times (X_n \to Y_{n+1}) \to Y_n$ be a family of continuous maps. If for every $r \in R$ there is at most one sequence $L_n(r) \in Y_n$ such that

$$L_n(r) = B_n(\lambda \alpha.r, \lambda x.L_{n+1}(r)),$$

then there is at most one family of continuous functions $h_n \colon P_n \to Y_n$ such that

$$h_n(p) = B_n(p, \lambda x.h_{n+1}(p_x)),$$

which automatically satisfy $h_n(\lambda \alpha.r) = L_n(r)$.

Definition 5.4.3. We refer to the system of equations $h_n(p) = B_n(p, \lambda x.h_{n+1}(p_x))$ as a specification of h_n by *bar recursion*.

The advantage of this recursion scheme is that it has only one equation and hence avoids the non-continuous case distinction discussed above. Notice that we do not require that the conditions of Lemma 5.4.2 hold, and hence a specification by bar recursion can have zero, one or more continuous solutions.

5.5. *The infinite product as the iteration of the binary product*

The functional equation that defines the functional $\Pi\colon (JD)^\omega \to JD^\omega$ in (Escardó, 2008, Section 8.1, page 30) is

$$\Pi(\varepsilon)(p) = x_0 * \Pi(\varepsilon')(p_{x_0}) \text{ where } x_0 = \varepsilon_0(\lambda x.p_x(\Pi(\varepsilon')(p_x))),$$

and where ε' is the sequence ε with its first term ε_0 removed, i.e. $\varepsilon'_i = \varepsilon_{i+1}$. This can be equivalently written as

$$\Pi(\varepsilon)(p) = x_0 * b(x_0),$$

where

$$\delta = \Pi(\varepsilon'), \qquad b(x) = \delta(\lambda\alpha.p(x * \alpha)), \qquad x_0 = \varepsilon_0(\lambda x.p(x * b(x))).$$

In turn, this can be written as

$$\Pi(\varepsilon) = \varepsilon_0 \otimes \Pi(\varepsilon'),$$

if, as in Section 3.3, we consider the variation of the finite product $\otimes$ that given two selections functions $\varepsilon \in JX_0$ and $\delta \in J\left(\prod_i X_{i+1}\right)$, produces $\varepsilon \otimes \delta \in J\left(\prod_i X_i\right)$. This variation is given by

$$(\varepsilon \otimes \delta)(p) = x_0 * b(x_0),$$

where b and x_0 are defined as above.

Remark 5.5.1. Equivalently, to define the variation of the binary product, we can consider the isomorphism $X_0 \times \prod_i X_{i+1} \to \prod_i X_i$ defined by $(x, \alpha) \mapsto x * \alpha$. In fact, because, as established in Section 6 below, J is a functor, we get an isomorphism $J\left(X_0 \times \prod_i X_{i+1}\right) \to J\left(\prod_i X_i\right)$. Then the original product $\varepsilon \otimes \delta \in J\left(X_0 \times \prod_i X_{i+1}\right)$ gives the above variation after the application of this last isomorphism:

$$JX_0 \times J\left(\prod X_{i+1}\right) \xrightarrow{\otimes} J\left(X_0 \times \prod_i X_{i+1}\right) \xrightarrow{\cong} J\left(\prod_i X_i\right).$$

It is thus natural to attempt to define $\bigotimes$, in the generality discussed in Section 5.1, as a solution to the functional equation $F(\varepsilon) = \varepsilon_0 \otimes F(\varepsilon')$. The above equations for Π make sense because it was assumed that $X_i = D$, for every i. But if we assume the type of F in the left-hand side of the above equation to be $\prod_i JX_i \to J\left(\prod_i X_i\right)$, then this forces the type of F in the right-hand side to be $\prod_i JX_{i+1} \to J\left(\prod_i X_{i+1}\right)$. Hence we instead consider the system of equations

$$F_n(\varepsilon) = \varepsilon_0 \otimes F_{n+1}(\varepsilon'),$$

with the continuous unknowns $F_n\colon \prod_i JX_{i+n} \to J\left(\prod_i X_{i+n}\right)$. We now show that if X_i and R are QCB spaces with R discrete, there a unique solution, using bar recursion.

Lemma 5.5.2. Assume that X_i and R are k-spaces with R discrete, and fix a sequence $\varepsilon_i \in JX_i$ of selection functions. The system of equations

$$\delta_n = \varepsilon_n \otimes \delta_{n+1}$$

with the unknowns $\delta_n \in J\left(\prod_{i+n} X_i\right)$ is equivalent to a specification of δ_n by bar recursion of the form

$$\delta_n(p) = B_n(p, \lambda x.\delta_{n+1}(p_x)).$$

Moreover, there is at most one solution, and if it exists then it satisfies

$$\delta_n(\lambda\alpha.r)(i) = \varepsilon_{i+n}(\lambda x.r).$$

Construction. Define $B_n \colon P_n \times (X_n \to Y_{n+1}) \to Y_n$ by

$$B_n(p, f) = x_n * f(x_n), \quad \text{where } x_n = \varepsilon_n(\lambda x.p(x * f(x))),$$

where $Y_n = \prod_i X_{i+n}$ and $P_n = (Y_n \to R)$.

Proof. Because $(P_n \to Y_n) = J\left(\prod_i X_{i+n}\right)$, we have that $\delta_n \colon P_n \to Y_n$, and

$$
\begin{aligned}
B_n(p, \lambda x.\delta_{n+1}(p_x)) &= x_n * b(x_n), \\
&\qquad \text{where } b(x) = \delta_{n+1}(p_x) \text{ and } x_n = \varepsilon_n(\lambda x.p(x * b(x))), \\
&= (\varepsilon_n \otimes \delta_{n+1})(p).
\end{aligned}
$$

Hence the equations $\delta_n(p) = B_n(p, \lambda x.\delta_{n+1}(p_x))$ are equivalent to $\delta_n = \varepsilon_n \otimes \delta_{n+1}$. Because $x_n = \varepsilon_n(\lambda x.r)$ if $p = \lambda\alpha.r$, the equations $L_n(r) = B_n(\lambda\alpha.r, \lambda x.L_{n+1}(r))$ amount to $L_n(r) = \varepsilon_n(\lambda x.r) * L_{n+1}(r)$. But there is a unique sequence $L_n(r) \in Y_n$ that satisfies this, namely $L_n(r)(i) = \varepsilon_{i+n}(\lambda x.r)$, and hence the result follows from Lemma 5.4.2. $\quad\square$

We emphasise that the next construction is not a specification by bar recursion, because a domain cannot be discrete, except in the uninteresting case it is the one-point space, and hence the two equations do not uniquely characterise δ_n. But if R is e.g. a lifted discrete space, this of course comes very close to a specification by bar recursion, which is what the proof of Theorem 5.5.4 exploits.

Lemma 5.5.3. If X_i and R are domains, then for every sequence of selection functions $\varepsilon_i \in JX_i$, there is a sequence of selection functions $\delta_n \in J\left(\prod_i X_{i+n}\right)$, continuously in ε, such that, for all n,

$$
\begin{aligned}
\delta_n(\lambda\alpha.r) &= \lambda i.\varepsilon_{i+n}(\lambda x.r), \\
\delta_n(p) &= B_n(p, \lambda x.\delta_{n+1}(p_x)),
\end{aligned}
$$

where B_n is defined in the construction of Lemma 5.5.2.

Proof. Let $F = \prod_n (P_n \to Y_n) = \prod_n J(Y_n)$, and define $H \colon F \to F$ by

$$H(h)(n)(p) = B_n(p, \lambda x.h_{n+1}(p_x)).$$

Then H is continuous and hence has a fixed point $\delta = \bigsqcup_k H^k(\bot)$ with $\delta_n \colon P_n \to Y_n$, because F is a domain. Then $\delta_n(p) = B_n(p, \lambda x.\delta_{n+1}(p_x))$ holds by construction. Moreover,

clearly B_n depends continuously on ε_n, and hence so do H and its least fixed point δ. By induction on k,

$$H^k(\bot)(n)(\lambda\alpha.r)(i) = \begin{cases} \varepsilon_{i+n}(\lambda x.r) & \text{if } i < k, \\ \bot & \text{if } i \geq k, \end{cases}$$

and hence $\delta_n(\lambda\alpha.r)(i) = \bigsqcup_k H^k(\bot)(n)(\lambda\alpha.r)(i) = \varepsilon_{i+n}(\lambda x.r)$, as claimed. $\qquad\square$

We do not know whether the following theorem holds more generally for k-spaces.

Theorem 5.5.4. If X_i and R are QCB spaces with R discrete, then for any sequence $\varepsilon_i \in JX_i$, there is a unique sequence $\delta_n = \delta_n(\varepsilon) \in J\left(\prod_i X_{i+n}\right)$ such that, for all n,

$$\delta_n = \varepsilon_n \otimes \delta_{n+1}.$$

Moreover, $\delta_n(\varepsilon)$ is continuous in ε.

Proof. We use the fact that QCB is fully and faithfully embedded into $\text{PER}(\omega\text{AlgLat})$ as described in (Bauer, 2002). The embedding transforms X_i and R into objects $(|X_i|, \sim_i)$ and $(|R|, \sim_R)$. It also gives Scott continuous functions $|\varepsilon_i|\colon (|X_i| \to |R|) \to |X_i|$ that preserve $\sim$, because the embedding preserves function spaces. Then we can apply Lemma 5.5.3 to the domains $|X_i|$ and $|R|$ under the Scott topology, and to the selection functions $|\varepsilon_i|$, to get selection functions $|\delta_n|$. Using the two equations of Lemma 5.5.3 as the base case and induction step of an argument by bar induction on p, one sees that for all n and $p \in P_n$, if $s_o, s_1\colon \prod_n |X_{i+n}| \to |R|$ track p, then $|\delta_n|(s_0) \sim |\delta_n|(s_1)$. Hence $s_0 \sim s_1$ implies $|\delta_n|(s_0) \sim |\delta_n|(s_1)$, and so $|\delta_n|$ is a morphism of $\text{PER}(\omega\text{AlgLat})$, which then gives a morphism δ_n of QCB, because the embedding of QCB into $\text{PER}(\omega\text{AlgLat})$ is full. $\quad\square$

Notice that the assumption of discreteness of R is used twice in this proof, so that bar induction can be applied to (1) establish that there is at most one solution in Lemma 5.5.2, and (2) prove totality of the functional constructed in Lemma 5.5.3. As discussed in Remark 5.5.8 below, such an assumption is essential.

Lemma 5.5.5. Under the assumptions and notation of Theorem 5.5.4, and additionally defining $\varepsilon_i^{(k)} := \varepsilon_{k+i}$ we have

$$\delta_n\left(\varepsilon^{(k)}\right) = \delta_{n+k}(\varepsilon).$$

Proof. If we apply Theorem 5.5.4 to the sequences ε and $\varepsilon^{(k)}$, we get sequences $\delta_n = \delta_n(\varepsilon)$ and $\zeta_n = \delta_n\left(\varepsilon^{(k)}\right)$ that satisfy $\delta_n = \varepsilon_n \otimes \delta_{n+1}$ and $\zeta_n = \varepsilon_n^{(k)} \otimes \zeta_{n+1} = \varepsilon_{n+k} \otimes \zeta_{n+1}$. But the sequence δ_{n+k} also satisfies the second equation, and hence, by uniqueness, $\zeta_n = \delta_{n+k}$, which amounts to the statement of the lemma. $\quad\square$

Corollary 5.5.6. The equation $F_n(\varepsilon) = \varepsilon_0 \otimes F_{n+1}(\varepsilon')$ has a unique continuous solution $F_n\colon \prod_i JX_{i+n} \to J\left(\prod_i X_{i+n}\right)$, namely $F_n(\varepsilon) = \delta_0\left(\varepsilon^{(n)}\right)$.

Proof. This equation amounts to $\delta_0(\varepsilon^{(n)}) = \varepsilon_0 \otimes \delta_0(\varepsilon^{(n+1)})$, which in turn amounts to $\delta_n(\varepsilon) = \varepsilon_0 \otimes \delta_{n+1}(\varepsilon)$ by Lemma 5.5.5, and holds uniquely by Theorem 5.5.4. $\quad\square$

Definition 5.5.7. For any sequence $\varepsilon_i \in JX_i$, we write $\bigotimes_i \varepsilon_i := \delta_0(\varepsilon)$. Then, by Corollary 5.5.6, this is characterised as

$$\bigotimes_i \varepsilon_i = \varepsilon_0 \otimes \bigotimes_i \varepsilon_{i+1}.$$

Remark 5.5.8. The assumption that R be discrete is essential. If $R = (\mathbb{N} \to \mathbb{N})$ and $X_i = \mathbb{N}$, for instance, we could take $p(\alpha)(m) = \alpha(m) + 1$ and $\varepsilon_n(q) = q(0)(n+1)$. In this case our equation would imply

$$\delta_0(p)(0) = x_0 = \varepsilon_0(\lambda x.p(x * b(x))) = p(0 * b(0))(1) = b(0)(0) + 1 = \delta_1(p_0)(0) + 1,$$

and by induction $\delta_0(p)(0) = \delta_n(p_{0^n})(0) + n$, for all n, and hence there is no solution δ_n for the equation of Theorem 5.5.4. This is adapted from a similar counter-example in (Berger and Oliva, 2005). Note, however, that for specific families ε_n and R non-discrete, a solution may exist, e.g. if ε_n are constant functions. But notice that, by virtue of the previous counter-example, the equation of Corollary 5.5.6 cannot have a solution if $R = (\mathbb{N} \to \mathbb{N})$.

5.6. *Infinite products of quantifiers*

We now proceed to the definition of infinite products of quantifiers, which turns out to be subtler. Any such notion ought to satisfy, in particular,

$$\bigotimes_i \exists_{X_i} = \exists_{\prod_i X_i}, \quad \bigotimes_i \forall_{X_i} = \forall_{\prod_i X_i}, \quad \bigotimes_i \sup_{X_i} = \sup_{\prod_i X_i}, \quad \bigotimes_i \inf_{X_i} = \inf_{\prod_i X_i}$$

when these quantifiers exist for suitable choices of R. We have seen that the selection function $\bigotimes_i \varepsilon_i$ is continuous in the sequence ε. However, such a quantifier $\bigotimes_i \phi_i$ cannot be continuous in ϕ. In fact, consider the particular case in which ϕ_i is the boolean-valued, bounded existential quantifier $\exists_{S_i}$ for a finite subset S_i of X_i. Then $\prod_i S_i$ is empty if and only if S_i is empty for some i. But an empty set may be present arbitrarily far away in the sequence S_i, and hence $(\bigotimes_i \exists_{S_i})(p)$ depends on the whole sequence $\exists_{S_i}$, which violates continuity. In connection with this, we observe, for future reference, that the bounded existential quantifier of the empty set is not attainable.

It is thus natural to attempt to define infinite products of quantifiers $\phi_i \in KX_i$ by mimicking Theorem 5.5.4 but giving up continuity of the formation of the product. Consider the system of equations

$$\gamma_n = \phi_n \otimes \gamma_{n+1}$$

with the unknowns $\gamma_n \in K\left(\prod_i X_{n+i}\right)$. This system of equations can be put in bar recursive form

$$\gamma_n(p) = B_n(p, \lambda x.\gamma_{n+1}(p_x)),$$

for a suitable B_n. In fact, because

$$
\begin{aligned}
\gamma_n(p) &= (\phi_n \otimes \gamma_{n+1})(p) = \phi_n(\lambda x.\gamma_{n+1}(\lambda \alpha.p(x * \alpha))) \\
&= \phi_n(\lambda x.\gamma_{n+1}(p_x))
\end{aligned}
$$

we can (and are forced to) define

$$B_n(p, f) = \phi_n(f).$$

Now the equation $L_n(r) = B_n(\lambda\alpha.r, \lambda x.L_{n+1}(r))$ amounts to

$$L_n(r) = \phi_n(\lambda x.L_{n+1}(r)).$$

If e.g. $X_i = R = \mathbb{N}$ and $\phi_i(q) = q(0) + 1$, then this equation reduces to $L_n(r) = L_{n+1}(r) + 1$, and, by induction, $L_n(r) = L_{n+k}(r) + k$ for every k, which is impossible and hence shows that there is no sequence γ_n such that $\gamma_n = \phi_n \otimes \gamma_{n+1}$. Thus, in general, the infinite iteration of the finite product of quantifiers fails to exist.

Now, by the discussion that follows Definition 3.0.4, if the quantifiers ϕ_n are attainable, then $\phi_n(\lambda x.r) = r$, and hence the above constraint on the sequence L_n amounts to $L_n(r) = L_{n+1}(r)$, i.e. L_n can be any constant sequence. Then Lemma 5.4.2 is not applicable, and, moreover, even if the equation $\gamma_n = \phi_n \otimes \gamma_{n+1}$ has a solution, all it says about $\gamma_n(\lambda\alpha.r)$ is that it must be a constant sequence $L_n(r)$. But if γ_n itself is required to be attainable, then $\gamma_n(\lambda\alpha.r) = r$. Hence, by bar induction, using this for the base case and $\gamma_n(p) = B_n(p, \lambda x.\gamma_{n+1}(p_x))$ for the inductive step, we see that if the quantifiers ϕ_i are attainable, then the system of equations $\gamma_n = \phi_n \otimes \gamma_{n+1}$ has at most one attainable solution.

Theorem 5.6.1. For every sequence of attainable quantifiers $\phi_i \in JX_i$, there is a unique sequence of attainable quantifiers $\gamma_n \in J\left(\prod_i X_{i+n}\right)$ such that, for all n,

$$\gamma_n = \phi_n \otimes \gamma_{n+1}.$$

Proof. It remains to establish existence. Let ε_i be a selection function for the quantifier ϕ_i. By Theorem 5.5.4, there is a unique sequence δ_i such that $\delta_n = \varepsilon_n \otimes \delta_{n+1}$. Taking $\gamma_n = \overline{\delta_n}$, Theorem 3.1.5 gives $\gamma_n = \overline{\varepsilon_n \otimes \delta_{n+1}} = \overline{\varepsilon_n} \otimes \overline{\delta_{n+1}} = \phi_n \otimes \gamma_{n+1}$, as required. $\square$

If we define $\bigotimes_i \phi_i = \gamma_0$, under the assumptions of this theorem, then, by construction,

$$\bigotimes_i \phi_i = \phi_0 \otimes \bigotimes_i \phi_{i+1}, \qquad \overline{\bigotimes_i \varepsilon_i} = \bigotimes_i \overline{\varepsilon_i}.$$

In particular, we do have that, as required above, for suitable choices of R discrete,

$$\bigotimes_i \exists_{X_i} = \exists_{\prod_i X_i}, \quad \bigotimes_i \forall_{X_i} = \forall_{\prod_i X_i}, \quad \bigotimes_i \sup_{X_i} = \sup_{\prod_i X_i}, \quad \bigotimes_i \inf_{X_i} = \inf_{\prod_i X_i},$$

provided the quantifiers of the left-hand sides of the equations exist and are attainable.

Question 5.6.2. By (Escardó, 2008, Theorem 6.3), if $R = \mathbb{B}$, if X is a space of Kleene–Kreisel functionals, and if $\emptyset \neq S \subseteq X$ has a quantifier $\exists_S \in KX$, then $\exists_S$ has a selection function continuously in $\exists_S$. By (Escardó, 2008, Lemma 5.5), such a set S has a quantifier $\exists_S \in KX$ if and only if it is compact. We remark that the universal quantifier $\forall_S$ is continuously interdefinable with the quantifier $\exists_S$, and that $\exists_S = \sup_S$ and $\forall_S = \inf_S$ for the case $R = \mathbb{B}$ with false $<$ true. We highlight the following open question. Under which constraints on R, X and $\phi \in KX$ do quantifiers ϕ have a selection function

continuously in ϕ? Notice that, for such quantifiers, the infinite product functional is continuous.

5.7. *Relation to traditional instances of bar recursion*

Recall that in Lemma 3.2.2 we developed a recursive characterisation of the optimal play

$$\left(\bigotimes_{i=0}^{n-1} \varepsilon_i \right)(p).$$

Before we proceed, let us consider an alternative recursive characterisation in which the sequence of selection functions ε and the outcome predicate p do not change in recursive calls. For this, we introduce a finite sequence s which is prefixed to the argument of p, and grows in recursive calls. Intuitively, each recursive call extends a partial play s of length k in an optimal way from k onwards to get a complete play. In fact, bar recursion is normally presented in this way, with the sequence s, rather than as in Section 5.4.

Definition 5.7.1. For $\varepsilon_i \in JX_i$ and $p \colon \prod_{i=0}^{n-1} X_i \to R$ fixed, and for each $k < n$, define

$$\mathrm{cbr}_k \colon \prod_{i=0}^{k-1} X_i \to \prod_{i=0}^{n-1} X_i$$

by

$$(\mathrm{cbr}_k(s))_i := \begin{cases} s_i & \text{if } i < k, \\ \varepsilon_i(\lambda x_i.p(\mathrm{cbr}_{i+1}(s * t * x_i))) & \text{if } n > i \geq k, \end{cases}$$

where $t = (\mathrm{cbr}_k(s))_k * \cdots * (\mathrm{cbr}_k(s))_{i-1}$. Notice that the equation $\mathrm{cbr}_n(s) = s$ is included in the above scheme, which gives a base case for the recursion.

Proposition 5.7.2. The family of functions $\mathrm{cbr}_k \colon \prod_{i=0}^{k-1} X_i \to \prod_{i=0}^{n-1} X_i$ is related to the product of selection functions as

$$\mathrm{cbr}_k(s) = s * \left(\bigotimes_{i=k}^{n-1} \varepsilon_i \right)(p_s).$$

In particular we have

$$\left(\bigotimes_{i=0}^{n-1} \varepsilon_i \right)(p) = \mathrm{cbr}_0().$$

The above family of functions $\mathrm{cbr}_k(s)$ can be viewed as a finite approximation to the functional

$$\mathrm{cbr} \colon \sum_k \prod_{i=0}^{k-1} X_i \to \prod_{i=0}^{\infty} X_i$$

which computes an infinite optimal play, from a partial play s of finite but unbounded length, as

$$\mathrm{cbr}(s)(i) := \begin{cases} s_i & \text{if } i < |s| \\ \varepsilon_i(\lambda x_i.q(\mathrm{cbr}(s * t * x_i))) & \text{if } i \geq |s|, \end{cases}$$

where $t = \text{cbr}(s)(|s|) * \cdots * \text{cbr}(s)(i-1)$ and $q \colon \Pi_{i=0}^{\infty} X_i \to R$. Note that we no longer have a fixed stopping point n, but if q is assumed to be a continuous functional, for instance, then we eventually reach a point of continuity of q and the bar recursion stops.

This functional cbr is very similar to (yet different from) the functional used by Berardi, Bezem and Coquand (Berardi et al., 1998), taking $s \colon \prod_{i \in \mathbb{N}} X_i$ with finite support,

$$\text{bbc}(s)(i) \overset{X_i}{:=} \begin{cases} s_i & \text{if } i \in \mathsf{dom}(s) \\ \varepsilon_i(\lambda x^{X_i}.q(\text{bbc}(s * (i, x)))) & \text{if } i \notin \mathsf{dom}(s). \end{cases}$$

Other instances of bar recursion include *Spector's bar recursion* (Spector, 1962), with $\varepsilon_k \colon (X_k \to \Pi_i X_i) \to \Pi_i X_i$,

$$\text{sbr}(s) \overset{\Pi_i X_i}{:=} \begin{cases} \widehat{s} & \text{if } q(\widehat{s}) < |s| \\ \varepsilon_{|s|}(\lambda x^{X_{|s|}}. \text{sbr}(s * x)) & \text{if } q(\widehat{s}) \geq |s|, \end{cases}$$

where $\widehat{(\cdot)}$ is any fixed mapping $\Sigma_k \Pi_{i=0}^{k-1} X_i \to \Pi_i X_i$, and *modified bar recursion* (Berger and Oliva, 2006), with $X_i = X$, for a fixed X,

$$\text{mbr}(s)(i) \overset{X}{:=} \begin{cases} s_i & \text{if } i < |s| \\ \varepsilon_i(\lambda x^X.q(\text{mbr}(s * x))) & \text{if } i \geq |s|. \end{cases}$$

6. The Continuation and Selection Monads

Crucial parts of the above development follow naturally from conceptual reasons expressed in terms of category theory. The above construction K is part of a well-known monad, known as the *continuation monad*, which we review here. We show that also J is part of a monad, which we refer to as the *selection monad*. The two monads are strong, which explains products of quantifiers and of selection functions in a unified way. Moreover, the procedure $\varepsilon \mapsto \bar{\varepsilon}$ that transforms selections functions into quantifiers given in Definition 3.0.6 is a monad morphism $J \to K$. This explains our main Theorem 3.1.5 that shows that attainable quantifiers are closed under finite products.

6.1. *Strong monads on cartesian closed categories*

Recall that a monad (Mac Lane, 1971) on a category $\mathcal{X}$ is a triple (T, η, μ) where $T \colon \mathcal{X} \to \mathcal{X}$ is a functor, and $\eta_X \colon X \to TX$ (the unit) and $\mu_X \colon TTX \to TX$ (the multiplication) are natural transformations, subject to the three equations

$$\begin{aligned} \mu_X \circ \eta_{TX} &= \text{id}_{TX} = \mu_X \circ T\eta_X \quad \text{(unit laws)}, \\ \mu_X \circ T\mu_X &= \mu_X \circ \mu_{TX} \quad\quad\;\; \text{(associativity law)}. \end{aligned}$$

The associativity law is fairly laborious and space consuming to check directly for the cases $T = J$ and $T = K$, because it involves three applications of T, which amounts to a nesting of six function spaces. In such situations, as is well known, it is often more convenient to derive the monad from a suitable adjunction (Mac Lane, 1971, page 134).

Assuming that the underlying category has finite products, the monad is strong if and

only if it admits a (necessarily unique) natural transformation

$$t_{X,Y} \colon X \times TY \to T(X \times Y)$$

subject to certain equations, which can be safely omitted because we work with the following characterisation: Further assuming that the category is cartesian closed, the monad is strong if and only if the functor is $\mathcal{X}$-enriched (Kock, 1970a). This means that its action on morphisms is tracked by a morphism

$$(X \to Y) \to (TX \to TY)$$

of $\mathcal{X}$, rather than merely a function from the hom-set $\mathcal{X}(X, Y)$ to the hom-set $\mathcal{X}(TX, TY)$. For example, in a cartesian closed category of continuous functions, this means that the assignment $f \mapsto Tf$ is continuous in f. When T is $\mathcal{X}$-enriched, the strength is given by the λ-definition

$$t_{X \times Y}(x, v) = T(\lambda y.(x, y))(v),$$

and automatically satisfies the alluded equations. Notice that $\lambda y.(x, y) \colon Y \to X \times Y$ and hence $T(\lambda y.(x, y)) \colon TY \to T(X \times Y)$.

Definition 6.1.1. Let T be a strong monad on a cartesian closed category $\mathcal{X}$. One defines a morphism

$$m_{X,Y} \colon TX \times TY \to T(X \times Y)$$

by

$$m_{X \times Y}(u, v) = \mu_{X \times Y}(T(\lambda x.t(x, v)))(u).$$

That is, given any fixed $v \colon TY$, we have $\lambda x.t(x, v) \colon X \to T(X \times Y)$; applying the functor T to this we get a map $TX \to TT(X \times Y)$, and composing with the multiplication $\mu_{X \times Y} \colon TT(X \times Y) \to T(X \times Y)$ we get the a map $TX \to T(X \times Y)$, which we apply to $u \colon TX$ to get $m_{X \times Y}(u, v)$.

Remarks 6.1.2.

1   This standard morphism makes T into a *monoidal monad* (Kock, 1972). This amounts to saying that

$$m_{1,X}(\eta_1(), u) \cong u \cong m_{X,1}(u, \eta_1())$$

and

$$m_{X,Y \times Z}(u, m_{Y,Z}(v, w)) \cong m_{X \times Y,Z}(m_{X,Y}(u, v), w)$$

via the isomorphisms

$$T(1 \times X) \cong TX \cong T(X \times 1), \qquad T(X \times (Y \times Z)) \cong T((X \times Y) \times Z).$$

2   We have defined m from t. We can recover t from m by

$$t(x, v) = m(\eta(x), v).$$

3   Any monad morphism $\theta_X \colon TX \to T'X$ commutes with the standard monoidal monad structure defined in 6.1.1:

$$\theta_{X \times Y}(m(u, v)) = m(\theta_X(u), \theta_Y(v)).$$

Recall that a monad morphism $T \to T'$ is a natural transformation $T \to T'$ that commutes with the functors, units and multiplications that define the monads.

6.2. *The continuation monad*

The continuation monad $KX = ((X \to R) \to R)$ is well known (Kock, 1970b; Moggi, 1990; Moggi, 1991), and hence we provide the constructions but omit a verification of the axioms. The easiest way to derive the continuation monad is by considering the functor $P \colon \mathcal{X} \to \mathcal{X}^{\mathrm{op}}$ defined by

$$PX := (X \to R).$$

This is an $\mathcal{X}$-enriched functor, since its action $Pf = (q \mapsto q \circ f)$ on morphisms,

$$P(X \xrightarrow{f} Y) = \left((Y \xrightarrow{q} R) \mapsto (X \xrightarrow{f} Y \xrightarrow{q} R) \right),$$

is tracked by a morphism $(X \to Y) \to (PY \to PX)$ of $\mathcal{X}$. This functor is self-adjoint on the right,

$$\mathcal{X}(A, PX) \cong \mathcal{X}(X, PA),$$

and the adjunction induces the continuation monad $K = PP$.

For a morphism $f \colon X \to Y$, the morphism $Kf \colon KX \to KY$ is given by

$$Kf(\phi)(q) = \phi(\lambda x.q(f(x))).$$

Because $f \mapsto Kf$ is λ-definable, it is a morphism of the category and hence the monad is strong, with strength $t_{X,Y} \colon X \times KY \to K(X \times Y)$ given by

$$
\begin{aligned}
t_{X,Y}(x, \gamma) &= K(\lambda y.(x, y))(\gamma) \\
&= \lambda p.\gamma(\lambda y.p(x, y)).
\end{aligned}
$$

The unit $\eta_X \colon X \to KX$ is defined by

$$\eta_X(x)(p) = p(x).$$

The multiplication $\mu_X \colon KKX \to KX$ is defined by

$$\mu(\Phi)(p) = \Phi(\lambda \phi.\phi(p)).$$

Remark 6.2.1. It is easily verified that the morphism $m_{X,Y} \colon KX \times KY \to K(X \times Y)$ defined in 6.1.1 satisfies

$$m_{X,Y}(\phi, \gamma) = \phi \otimes \gamma,$$

where $\otimes$ is defined in Section 2.1, and hence $t_{X,Y}(x, \gamma) = \eta(x) \otimes \gamma$. By Remark 6.1.2, we conclude that the product of quantifiers is associative: $(\phi_0 \otimes \phi_1) \otimes \phi_2 \cong \phi_0 \otimes (\phi_1 \otimes \phi_2)$.

We now illustrate the meaning of these constructions in the context of this work.

Examples 6.2.2. Consider $R = \Omega$ in the category of sets or any topos.

1 For any $A \subseteq X$ and $f \colon X \to Y$, the bounded quantifiers $\exists_A, \forall_A \in KX$ and $\exists_{f(A)}, \forall_{f(A)} \in KY$ satisfy

$$Kf(\exists_A) = \exists_{f(A)}, \qquad Kf(\forall_A) = \forall_{f(A)},$$

because $\exists y \in f(A)(q(y)) = \exists x \in A(q(f(x)))$ and similarly for $\forall$. In this sense, Kf behaves like an f-image operator, and functoriality says that the g-image of the f-image is the same as the $(g \circ f)$-image.

2   For the strength $t_{X,Y} \colon X \times KY \to K(X \times Y)$, we have, for any $x \in X$ and any $B \subseteq Y$,

$$t(x, \exists_B) = \exists_{\{x\} \times B}, \qquad t(x, \forall_B) = \forall_{\{x\} \times B}.$$

Similarly, by Example 2.1.2, for the operation $m_{X,Y} \colon KX \times KY \to K(X \times Y)$, we have for any $A \subseteq X$,

$$m(\exists_A, \exists_B) = \exists_A \otimes \exists_B = \exists_{A \times B}, \qquad m(\forall_A, \forall_B) = \forall_A \otimes \forall_B = \forall_{A \times B}.$$

3   The unit produces the bounded existential/universal quantifier for the singleton set:

$$\eta_X(x) = \exists_{\{x\}} = \forall_{\{x\}},$$

because $\eta_X(x)(p) = p(x) = \exists x \in \{x\}(p(x)) = \forall x \in \{x\}(p(x))$.

4   The multiplication $\mu_X \colon KKX \to KX$ involves the perhaps unfamiliar concept of quantification over quantifiers. Suppose $A \subseteq KX$ is a set such that each $\phi \in A$ is the bounded existential quantifier of a set $B_\phi \subseteq X$, i.e.

$$\phi = \exists_{B_\phi}$$

Then the bounded universal quantifier $\forall_A \in KKX$ of the set $A \subseteq KX$ satisfies

$$\mu(\forall_A)(p) = \forall \phi \in A \, \exists x \in B_\phi(p(x)).$$

6.3. *The selection monad*

To prove that J is a monad, we construct a new category, which will turn out to be the Kleisli category of J, and show that there is an adjunction with $\mathcal{X}$. In order to define this manifestation of the Kleisli category of J, we simultaneously work with a manifestation of the Kleisli category of K.

We have used letters X, Y, Z for the objects of our underlying category $\mathcal{X}$. In order to both avoid confusion and be compatible with the notational conventions of MacLane (Mac Lane, 1971) for the objects of two different categories related by an adjunction, we will now also adopt the letters A, B, C. These new letters will stand for objects of a Kleisli category, or equivalently the category of free algebras. Similarly, in an adjoint situation, we will use the letter f for morphisms of $\mathcal{X}$ and the letter g for morphisms of free algebras.

A morphism $A \to B$ of the Kleisli category of K is a morphism $A \to KB$ of the underlying category $\mathcal{X}$, which, by transposition, amounts to a morphism

$$(B \to R) \to (A \to R).$$

For the proof that J is a monad that we are about to develop, it is convenient to abstract from this situation, by considering an arbitrary functor

$$P \colon \mathcal{X} \to \mathcal{X}^{\mathrm{op}}$$

and hence morphisms of the form

$$PB \to PA.$$

We will recover the intended results by considering $PX = (X \to R)$ as in Section 6.2.

For the remainder of this section, let $P \colon \mathcal{X} \to \mathcal{X}^{\mathrm{op}}$ be an enriched functor that is self-adjoint on the right, in the sense that there is a natural isomorphism

$$\mathcal{X}(X, PA) \cong \mathcal{X}(A, PX).$$

We remark that the following definition does not require that the functor P be enriched or self-adjoint on the right, but everything else does. The enrichment is needed in order to be able to define new morphisms and enriched functors using the lambda-calculus.

Definition 6.3.1. Define a new category $\mathcal{K}$ from our underlying category $\mathcal{X}$ and the functor $P \colon \mathcal{X} \to \mathcal{X}^{\mathrm{op}}$ as follows:

1 *Objects of $\mathcal{K}$:* the same as those of $\mathcal{X}$.
2 *Morphisms of $\mathcal{K}$:* A morphism $f \colon A \to B$ of $\mathcal{K}$ is a morphism $f \colon PB \to PA$ of $\mathcal{X}$:

$$\mathcal{K}(A, B) = \mathcal{X}(PB, PA).$$

3 *Composition of $\mathcal{K}$:* For $f \colon A \to B$ and $g \colon B \to C$ in $\mathcal{K}$, that is, $f \colon PB \to PA$ and $g \colon PC \to PB$ in $\mathcal{X}$, define

$$g \,\square\, f = f \circ g.$$

4 *Identities of $\mathcal{K}$:* Of course, the identity of A in $\mathcal{K}$ is the identity of PA in $\mathcal{X}$.

Notice that, in the following lemma, both adjunctions (P, P) and (F, G) induce the same monad on $\mathcal{X}$, namely $K = PP$, and that, by construction, $\mathcal{K}$ is isomorphic to the Kleisli category $\mathcal{X}_K$.

Lemma 6.3.2. The functor $F = F_K \colon \mathcal{X} \to \mathcal{K}$ that is the identity on objects and sends $f \colon X \to Y$ to $Pf \colon PY \to PX$, regarded as a morphism $X \to Y$ of $\mathcal{K}$, has a right adjoint $G = G_K \colon \mathcal{K} \to \mathcal{X}$.

Proof. On objects, $GA = PPA$, and G sends a morphism $g \colon A \to B$ of $\mathcal{K}$, regarded as a morphism $g \colon PB \to PA$ of $\mathcal{X}$, to $Pg \colon PPA \to PPA$. By construction, a natural isomorphism $\mathcal{K}(FX, A) \cong \mathcal{X}(X, GA)$ amounts to a natural isomorphism $\mathcal{X}(X, PA) \cong \mathcal{X}(A, PX)$, which is given by assumption that P is self-adjoint on the right. $\square$

A morphism $A \to B$ of the Kleisli category of J is a morphism $A \to JB$ of the underlying category, which, by transposition, amounts to a morphism

$$(B \to R) \to (A \to B).$$

For our proof that J is a monad, we abstract from this situation as above, and consider morphisms of the form

$$PB \to (A \to B).$$

Definition 6.3.3. Define a new category $\mathcal{J}$ from our underlying category $\mathcal{X}$ and the enriched functor $P \colon \mathcal{X} \to \mathcal{X}^{\mathrm{op}}$ as follows:

1   *Objects of $\mathcal{J}$:* the same as those of $\mathcal{X}$.
2   *Morphisms of $\mathcal{J}$:* A morphism $f\colon A \to B$ of $\mathcal{J}$ is a morphism $f\colon PB \to (A \to B)$ of $\mathcal{X}$:

$$\mathcal{J}(A, B) = \mathcal{X}(PB, A \to B).$$

We think of such a morphism as a kind of parametrised morphism of $\mathcal{X}$, and we write the parameter as a subscript: For $f\colon PB \to (A \to B)$ and $q\colon PB$ and $a\colon A$, write

$$f_q(a) = f(q)(a).$$

3   *Auxiliary construction:* This parameter $q\colon PB$ can be "transferred back" to a new parameter $p\colon PA$ using $Pf_q\colon PB \to PA$:

$$p = Hf(q) := Pf_q(q)$$

4   *Composition of $\mathcal{J}$:* For

$$f\colon PB \to (A \to B), \qquad g\colon PC \to (B \to C)$$

in $\mathcal{X}$, define the composite

$$g \mathbin{\square} f\colon PC \to (A \to C)$$

by, for any $r\colon PC$,

$$(g \mathbin{\square} f)_r = g_r \circ f_{Hg(r)}.$$

That is, we compose functions in the usual way, but transferring back the parameter.
5   *Identities of $\mathcal{J}$:* The constantly identities $\mathrm{id}_q = \mathrm{id}$ of $\mathcal{X}$. It is clear that these act as left and right identities of composition, because $H\,\mathrm{id}(p) = p$.
6   *Associativity.* We first establish:

Claim: $H(g \mathbin{\square} f) = Hf \circ Hg$.
When we know that $\mathcal{J}$ is a category, this, together with the fact that $H\mathrm{id} = \mathrm{id}$, will amount to saying that H is a covariant functor $\mathcal{J} \to \mathcal{K}$ with object part $HA = A$, because $Hf \circ Hg = Hg \mathbin{\square} Hf$ by definition of composition in $\mathcal{K}$. But we will first use this claim to prove that $\mathcal{J}$ is a category.

Proof of the claim:

$$\begin{aligned} H(g \mathbin{\square} f)(r) &= P(g \mathbin{\square} f(r))(r) = P(g_r \circ f_{Hg(r)})(r) = Pf_{Hg(r)} \circ Pg_r(r) \\ &= Pf_{Hg(r)}(Hg(r)) = Hf(Hg(r)). \end{aligned}$$

Proof of associativity: Let

$$f\colon PB \to (A \to B), \qquad g\colon PC \to (B \to C), \qquad h\colon PD \to (B \to D),$$

and $s\colon PD$, and calculate:

$$\begin{aligned} ((h \mathbin{\square} g) \mathbin{\square} f))_s &= (h \mathbin{\square} g)_s \circ f_{H(h \mathbin{\square} g)(s)} = h_s \circ g_{Hh(s)} \circ f_{Hg(Hh(s))} \\ &= h_s \circ (g \mathbin{\square} f)_{Hh(s)} = (h \mathbin{\square} (g \mathbin{\square} f))_s. \end{aligned}$$

Hence $\mathcal{J}$ is indeed a category and $H\colon \mathcal{J} \to \mathcal{K}$ is a functor.

Notational warning. In the following proof, we use the letter epsilon for the counit, as is customary, but in its form ϵ. Recall that we also use the variant ε of the letter epsilon for selection functions, which in the following proof correspond to functions $PA \to A$.

Lemma 6.3.4. The functor $F = F_J \colon \mathcal{X} \to \mathcal{J}$ that is the identity on objects and that sends a morphism $f \colon X \to Y$ to the constantly f morphism $PY \to (X \to Y)$ of $\mathcal{X}$ has a right adjoint $G = G_J \colon \mathcal{J} \to \mathcal{X}$.

Proof. We describe G and the required natural isomorphim

$$\mathcal{J}(FX, A) \cong \mathcal{X}(X, GA)$$

by a universal property, appealing to (Mac Lane, 1971, Theorem IV-2(iv), page 81). It suffices to show that for every $\mathcal{J}$-object A there is a universal morphism from F to A. By definition, this amounts to saying that there are an $\mathcal{X}$-object GA and a $\mathcal{J}$-morphism $\epsilon = \epsilon_A \colon FGA \to A$ such that for every $\mathcal{J}$-morphism $f \colon FX \to A$, the equation $\epsilon \square Fg = f$ holds for a unique $\mathcal{X}$-morphism $g \colon X \to GA$. Considering $X = 1$ in the desired natural isomorphism, this suggests to choose GA to be

$$GA = (PA \to A).$$

We define $\epsilon \colon FGA \to A$ in $\mathcal{J}$, or equivalently $\epsilon \colon PA \to ((PA \to A) \to A)$ in $\mathcal{X}$, to be, for any $p \colon PA$ and $\varepsilon \colon (PA \to A)$,

$$\epsilon_p(\varepsilon) = \varepsilon(p).$$

For any $g \colon X \to GA$ in $\mathcal{X}$, that is, $g \colon X \to (PA \to A)$,

$$(\epsilon \square Fg)_p(x) = (\epsilon_p \circ (Fg)_{H\epsilon(p)})(x) = (\epsilon_p \circ g)(x) = \epsilon_p(g(x)) = g(x)(p).$$

Hence given any $f \colon FX \to A$ in $\mathcal{J}$, or equivalently, $f \colon PA \to (X \to A)$ in $\mathcal{X}$, we are forced to define $g \colon X \to GA$, by

$$g(x)(p) = f_p(x).$$

With this, not only does the equation $\epsilon \square Fg = f$ hold, but also it uniquely determines g, as required to conclude the existence of a right adjoint. $\square$

The above proof does not say explicitly how G acts on morphisms. By the proof of (Mac Lane, 1971, Theorem IV-2(iv), page 81), its action on morphisms is uniquely determined by its action on objects and the requirement that $\epsilon \colon FGA \to A$ be a natural transformation in A, as follows: Given a morphism $h \colon A \to B$ in $\mathcal{J}$, the morphism Gh in $\mathcal{X}$ is the unique $g \colon GA \to GB$ such that $\epsilon \square Fg = f$, where $f = h \square \epsilon \colon FGA \to B$. Now, by the construction of g from f in the proof of Lemma 6.3.4, expanding all the definitions, and using the fact that a morphism $h \colon A \to B$ of $\mathcal{J}$ is a morphism $h \colon PB \to (A \to B)$ of $\mathcal{X}$, we have that, for all $q \in PB$ and $\varepsilon \in GA = (PA \to A)$,

$$\begin{aligned} Gh(\varepsilon)(q) &= g(\varepsilon)(q) = f_q(\varepsilon) = (h \square \epsilon)_q(\varepsilon) = (h_q \circ \epsilon_{Hh(q)})(\varepsilon) \\ &= h_q(\varepsilon(Hh(q))) = h_q(\varepsilon(Ph_q(q))). \end{aligned}$$

The proof of Lemma 6.3.4 does not say explicitly what the unit $\eta_X\colon X \to GFX$ of the adjunction is either. By the proof of (Mac Lane, 1971, Theorem IV-2(iv), page 81), it is the unique morphism such that $\epsilon \,\square\, Fg = \eta_X$ for $g = \mathrm{id}_{FX}\colon FX \to FX$. By the construction of g from f in the proof of Lemma 6.3.4 and the definition of the identities of $\mathcal{J}$,

$$\eta_X(x)(p) = (\mathrm{id}_{FX})_p(x) = x.$$

Applying the standard construction of a monad from a given adjunction, we get:

Lemma 6.3.5. $JX = GFX = (PX \to X)$ is a strong monad on $\mathcal{X}$, with action on morphisms $f\colon X \to Y$ given by, for all $\varepsilon\colon JX$ and $q\colon PY$,

$$Jf(\varepsilon)(q) = f(\varepsilon((Pf(q)))),$$

and with units $\eta_X\colon X \to JX$ and multiplications $\mu_X\colon JJX \to JX$ given by, for all $x\colon X$, $p\colon PX$, and $E\colon JJX$,

$$\eta_X(x)(p) = x, \qquad\qquad \mu_X(E)(p) = E(P\epsilon_p(p))(p).$$

Proof. Let $f\colon X \to Y$ in $\mathcal{X}$ and $h = Ff$. Then $h_q = f$ by the definition of F, and hence we conclude that

$$GFf(q) = h_q(\varepsilon(Ph_q(q))) = f(\varepsilon(Pf(q))),$$

as claimed. The multiplication is given by, for any $E\colon JJX$ and $p\colon PX$,

$$\mu(E)(p) = G\epsilon(E)(p) = \epsilon_p(E(P\epsilon_p(p))) = E(P\epsilon_p(p))(p),$$

as claimed. The monad is strong because the action of J on morphisms is clearly λ-definable, and hence tracked by a morphism of $\mathcal{X}$. $\qquad\square$

Lemma 6.3.6. There is a monad morphism $\theta_X\colon JX \to KX$, given by the adjoint transposes of the family of maps

$$\lambda p.P(\lambda\varepsilon.\varepsilon(p))(p)\colon PX \to PJX.$$

Proof. We apply (Moggi, 1990, Proposition 4.0.10), which shows that monad morphisms $\theta\colon J \to K$ are in bijection with functors $H\colon \mathcal{X}_J \to \mathcal{X}_K$ of the Kleisli categories which are the identity on objects and such that the equation $H \circ F_J = F_K$ holds. The direction of the bijection that we need constructs the component $\theta_X\colon JX \to KX$ of the natural transformation as Hh, where the morphism h in $\mathcal{X}_J$ is the identity $JX \to JX$ of X in $\mathcal{X}$, regarded as a morphism $JX \to X$ of the Kleisli category $\mathcal{X}_J$ of J. In the manifestation $\mathcal{J}$ of $\mathcal{X}_J$, this amounts to a morphism $h\colon PX \to (JX \to X)$ of $\mathcal{X}$, which is readily seen to be $h_p(\varepsilon) = \varepsilon(p)$. Notice that $Hh\colon JX \to X$, because H is the identity on objects, and hence $Hh\colon JX \to KX$ regarded as a morphism of the Kleisli category of K, and hence $Hh\colon PX \to PJX$ regarded as a morphism of $\mathcal{X}$. Now, for $H\colon \mathcal{J} \to \mathcal{K}$ constructed in Definition 6.3.3, we have

$$H(F_J f)(q) = Pf(q) = F_K f(q),$$

and hence the above is applicable. We thus get θ as the adjoint transpose $JX \to PPX$ of

$$Hh = \lambda p.Ph_p(p) = \lambda p.P(\lambda \varepsilon.\varepsilon(p))(p)$$

which concludes the proof. $\qquad \square$

Theorem 6.3.7. $JX = ((X \to R) \to X)$ is a strong monad on $\mathcal{X}$, with action on morphisms $f \colon X \to Y$ given by

$$Jf(\varepsilon)(q) = f(\varepsilon(q \circ f)),$$

and with units $\eta_X \colon X \to JX$ and multiplications $\mu_X \colon JJX \to JX$ given by

$$\eta_X(x)(p) = x, \qquad \mu_X(E)(p) = E(\lambda \varepsilon.p(\varepsilon(p)))(p).$$

Moreover, the assignment $\varepsilon \to \bar{\varepsilon}$ is a monad morphism from J to the continuation monad $KX = ((X \to R) \to R)$.

Proof. Take $PX = (X \to R)$ and $Pf(q) = q \circ f$ as in Section 6.2. Then

$$P\epsilon_p(p) = p \circ \epsilon_p = \lambda \varepsilon.p(\epsilon_p(\varepsilon)) = \lambda \varepsilon.p(\varepsilon(p)),$$

which gives the above definition of μ. Now

$$\lambda p.P(\lambda \varepsilon.\varepsilon(p))(p) = \lambda p.\lambda \varepsilon.p(\varepsilon(p)),$$

whose transpose is $\lambda \varepsilon.\lambda p.p(\varepsilon(p))$ and hence gives the desired monad morphism. $\qquad \square$

Remarks 6.3.8.

1 Because a monad morphism commutes with the functors that define the monads, Theorem 6.3.7 gives, for any $f \colon X \to Y$ and $\varepsilon \in JX$:

$$\overline{Jf(\varepsilon)} = Kf(\bar{\varepsilon}).$$

Hence if $\varepsilon \in JX$ is a selection function for the quantifier $\phi \in KX$ then $Jf(\varepsilon)$ is a selection function for the image quantifier $Kf(\phi)$. In particular, by Example 6.2.2(1), for any $A \subseteq X$, if ε is a selection function for $\exists_A$, then $Jf(\varepsilon)$ is a selection function for $\exists_{f(A)}$, which is the content of the proof of (Escardó, 2008, Proposition 4.3).

2 Theorem 3.1.5 follows directly from Theorem 6.3.7 and Remarks 6.1.2.

3 The construction of the strength and of the monoidal monad structure given in Definition 6.1.1 is characterised as follows, where $\otimes$ is defined as in Section 3.1:

(a) The morphism $t_{X,Y} \colon X \times JY \to J(X \times Y)$ satisfies

$$t(x, \delta) = \lambda p.(x, \delta(\lambda y.p(x, y))) = \eta(x) \otimes \delta$$

(b) The morphism $m_{X,Y} \colon JX \times JY \to J(X \times Y)$ satisfies

$$m(\varepsilon, \delta) = \varepsilon \otimes \delta.$$

(c) Hence, by Remark 6.1.2, we conclude that the product of selection functions is associative: $(\varepsilon_0 \otimes \varepsilon_1) \otimes \varepsilon_2 \cong \varepsilon_0 \otimes (\varepsilon_1 \otimes \varepsilon_2)$.

7. Further work

The work presented here sets the ground for applications to proof theory which we are currently developing. We study the role of the monad J in the translation of proofs in the context of minimal logic ML, where monad algebras $JA \to A$ are objects with a realizer/proof of the instance

$$\mathsf{PL}_R A \quad : \quad ((A \to R) \to A) \to A$$

of Peirce's law. Also, in the same way the monad K gives rise to the well-known negative translation, the monad J defines a proof translation of $\mathsf{ML} + \mathsf{PL}_R$ into ML. We also know that the infinite product functional $\bigotimes$ realizes (in the sense of Kreisel's modified realizability) the J-shift

$$\forall n (JA(n)) \to J\left(\forall n A(n)\right).$$

The J-shift is more general than the double negation shift (K-shift), and gives the K-shift in the cases it exists, similarly to the relation between countable products of selection functions and quantifiers discussed in Section 5.6. This leads to a natural construction based on the product of selection functions which realises the axiom of countable (and dependent) choice.

We are also investigating the inter-definability (over Gödel's system T) between the new instance of bar recursion presented here and traditional instances (cf. Section 5.7).

References

Abramsky, S. and Jung, A. (1994). Domain theory. In Abramsky, S., Gabbay, D., and Maibaum, T., editors, *Handbook of Logic in Computer Science*, volume 3 of *Oxford science publications*, pages 1–168. Oxford university press.

Avigad, J. and Feferman, S. (1998). Gödel's functional ("Dialectica") interpretation. In Buss, S. R., editor, *Handbook of proof theory*, volume 137 of *Studies in Logic and the Foundations of Mathematics*, pages 337–405. North Holland, Amsterdam.

Battenfeld, I., Schröder, M., and Simpson, A. (2006). Compactly generated domain theory. *Math. Structures Comput. Sci.*, 16(2):141–161.

Battenfeld, I., Schröder, M., and Simpson, A. (2007). A convenient category of domains. In *Computation, meaning, and logic: articles dedicated to Gordon Plotkin*, volume 172 of *Electron. Notes Theor. Comput. Sci.*, pages 69–99. Elsevier, Amsterdam.

Bauer, A. (2002). A relationship between equilogical spaces and type two effectivity. *MLQ Math. Log. Q.*, 48(suppl. 1):1–15.

Bekič, H. (1984). *Programming languages and their definition*, volume 177 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin. Selected papers edited by C. B. Jones.

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

Berger, U. and Oliva, P. (2005). Modified bar recursion and classical dependent choice. *Lect. Notes Log.*, 20:89–107.

Berger, U. and Oliva, P. (2006). Modified bar recursion. *Math. Structures Comput. Sci.*, 16(2):163–183.

Bezem, M. (1985). Strongly majorizable functionals of finite type: a model for bar recursion containing discontinuous functionals. *The Journal of Symbolic Logic*, 50:652–660.

Bove, A. and Dybjer, P. (2008). Dependent types at work. Lecture notes from the LerNET Summer School, Piriapolis, available at the authors' web pages.

Escardó, M. (2007). Infinite sets that admit fast exhaustive search. In *Proceedings of the 22nd Annual IEEE Symposium on Logic In Computer Science*, pages 443–452. IEEE Computer Society.

Escardó, M. (2008). Exhaustible sets in higher-type computation. *Log. Methods Comput. Sci.*, 4(3):3:3, 37.

Escardó, M., Lawson, J., and Simpson, A. (2004). Comparing Cartesian closed categories of (core) compactly generated spaces. *Topology Appl.*, 143(1-3):105–145.

Hutton, G. (2007). *Programming in Haskell*. Cambridge University Press.

Johnstone, P. (2002). *Sketches of an Elephant: a Topos Theory Compendium*. Oxford University Press.

Kock, A. (1970a). Monads on symmetric monoidal closed categories. *Arch. Math. (Basel)*, 21:1–10.

Kock, A. (1970b). On double dualization monads. *Math. Scand.*, 27:151–165 (1971).

Kock, A. (1972). Strong functors and monoidal monads. *Arch. Math. (Basel)*, 23:113–120.

Lambek, J. and Scott, P. (1986). *Introduction to Higher Order Categorical Logic*. Cambridge University Press.

Mac Lane, S. (1971). *Categories for the Working Mathematician*. Springer.

Moggi, E. (1990). An abstract view of programming languages. Technical Report ECS-LFCS-90-113, Laboratory for Foundations of Computer Science, University of Edinburgh.

Moggi, E. (1991). Notions of computation and monads. *Information and Computation*, 93(1):55–92.

Nisan, N. et al. (2007). *Algorithmic Game Theory*. Cambridge University Press.

Normann, D. (1980). *Recursion on the countable functionals*, volume 811 of *Lec. Not. Math.* Springer.

Normann, D. (1999). The continuous functionals. In Griffor, E. R., editor, *Handbook of Computability Theory*, chapter 8, pages 251–275. North Holland, Amsterdam.

Oliva, P. (2006). Understanding and using Spector's bar recursive interpretation of classical analysis. Beckmann, Arnold (ed.) et al., Logical approaches to computational barriers. Second conference on computability in Europe, CiE 2006, Swansea, UK, June 30–July 5, 2006. Proceedings. Berlin: Springer. Lecture Notes in Computer Science 3988, 423-434 (2006).

Schröder, M. (2002). Extended admissibility. *Theoret. Comput. Sci.*, 284(2):519–538.

Smyth, M. (1977). Effectively given domains. *Theoret. Comput. Sci.*, 5(1):256–274.

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

Valiente, G. (2002). *Algorithms on Trees and Graphs*. Springer.