Math. Struct. in Comp. Science (2010), *vol.* 20, *pp.* 127–168. © Cambridge University Press 2010
doi:10.1017/S0960129509990351

Selection functions, bar recursion and backward induction

MARTÍN ESCARDÓ[†] and PAULO OLIVA[‡§]

[†]*University of Birmingham, Birmingham B15 2TT, U.K.*
Email: m.escardo@cs.bham.ac.uk
[‡]*Queen Mary University of London, London E1 4NS, U.K.*
Email: paulo.oliva@eecs.qmul.ac.uk

Received 2 July 2009; revised 11 November 2009

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

In this paper 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 both the number of players and who plays at each round. The outcome of a game is specified by an n-ary predicate $p : \Pi_{i=0}^{n-1} X_i \to R$, 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, assuming the aim at each round is to force the outcome to be the value true, the existence of a winning strategy corresponds to the satisfiability of the predicate

§ The second author gratefully acknowledges the support of the Royal Society under grant 516002.K501/RH/kk.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

p, that is,

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

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, that is,

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

Now consider games with more than two outcomes, for example, $R = \{-1, 0, 1\}$. 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

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

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 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).

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

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

which we leave open in the definition of the game. When ϕ_i are the standard quantifiers $\exists, \forall : (X \to \mathbb{B}) \to \mathbb{B}$ or the supremum and infimum functionals $\sup, \inf : (X \to R) \to R$, 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 $\phi : (X \to R) \to R$ have *selection functions*, that is, functions

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

satisfying $\phi(p) = p(\varepsilon(p))$. For example, a selection function for the supremum functional $\sup : (X \to R) \to R$, 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 the first author's paper Escardó (2008). This is explained in Section 5, where we also show how the infinite product amounts to a form of bar recursion.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 selection 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 of the paper*

In Section 2 we discuss generalised quantifiers, finite products of quantifiers and sequential games. Section 3 covers selection functions, finite products of selection functions and the calculation of optimal strategies. Section 4 describes some applications. In Section 5 we consider infinite products of selection functions and quantifiers, and bar recursion, and in Section 6, the continuation and selection monads. In Section 7 we discuss some further work based on the work in this paper.

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 to familiarise themselves with such concepts. These sections are formally developed in the generality of cartesian closed categories, but can be read 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 said to be 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 in order to 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 topological spaces, $(X \to Y)$ is the set of continuous maps with a suitable topology (see, for example, 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 majorisable functionals (Bezem 1985);
(iii) spaces 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 and QCB spaces (Battenfeld *et al.* 2007), Kleene–Kreisel spaces and continuous functionals (Normann 1980) and various categories of domains under the Scott topology (Abramsky and Jung 1994);
(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 realisability toposes, among others.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

When working with our underlying cartesian closed category, we reason with generalised elements and the λ-calculus. So, for example, for any given $m : X \times X \to X$, 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 $x : X$, 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, for example, 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*. We 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 2.1.

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 (the principle of excluded middle and the axiom of choice).
3 Continuing from the above, we define

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

for $p : X \times Y \to R$. Then $\phi \in K(X \times Y)$.
4 We assume R is the real line $\mathbb{R}$ in a cartesian closed category of spaces and continuous functions (such as k-spaces, QCB spaces, and so on). We define

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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

5 Continuing from the previous example, we define

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

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

2.1. *Finite products of quantifiers*

The above Examples 2.1(3) and 2.1(5) are instances of the following construction.

Definition 2.2. Given quantifiers $\phi \in KX$ and $\gamma \in KY$, define a new quantifier

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

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

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

Examples 2.3.

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 $\exists_X \otimes \exists_Y = \exists_{X\times Y}$.

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

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 v_i be Borel regular measures on locally compact Hausdorff spaces X_i for $i = 0, 1$, and define $\phi_i \in KX_i$ by $\phi_i(p) := \int p dv_i$. Then $\phi \in K(X_0 \times X_1)$ defined by

$$\phi(p) := \int p d(v_0 \times v_1)$$

satisfies $\phi = \phi_0 \otimes \phi_1$, where $v_0 \times v_1$ is the product measure.

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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Definition 2.4. Given quantifiers $\phi \in \prod_{i=0}^{n-1} KX_i$, we define $\bigotimes_{i=0}^{n-1} \phi_i \in K(\prod_{i=0}^{n-1} X_i)$ 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 $\lambda x_0.(\bigotimes_{i=1}^{n-1} \phi_i)(p_{x_0})$. 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 are convenient for expressing some general notions regarding finite sequential games. It should be noted

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 2.5. Consider an alternating, two-person game that finishes after exactly n moves, with one of the players winning. The ith 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 says 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 is 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 expressed concisely as

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

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. However, we still require the game to have a fixed length n.

Definition 2.6. Let $(X_i)_{i=0}^{n-1}$ be an n-tuple of objects, $p\colon \prod_{i=0}^{n-1} X_i \to R$ be a predicate and $\phi\colon \prod_{i=0}^{n-1} K X_i$ 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 $k \leqslant n$, 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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 $w := (\bigotimes_{i=0}^{n-1} \phi_i)(p)$.
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 $k < n$,

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

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 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, that is, $w_{\vec{a}} = w_{\vec{a}*a_k}$.

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,

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

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

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

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

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

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

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 2.7. In Example 2.5, the optimal outcome w of the game says 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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

draw, and 1 that Eloise wins. We replace the existential and universal quantifiers by the minimum and maximum value functionals $\min_X, \max_X : (X \to R) \to R$ 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 $p : [0, 1] \to \mathbb{R}$ there is $a \in [0, 1]$ such that

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

Similarly, the maximum value theorem says that any continuous $p : X \to \mathbb{R}$ 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 $p : X \to R$, 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 $p : X \to R$ 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 both 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 the following stronger statement.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

(2) There is a function $\varepsilon : (X \to R) \to X$ 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 3.1. Given a quantifier $\phi : (X \to R) \to R$, any function $\varepsilon : (X \to R) \to X$ such that

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

for all $p : X \to R$, is called a *selection function for* ϕ. A quantifier that has a selection function is said to be *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 $\phi : (X \to R) \to R$ is attainable, 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 $\varepsilon : (X \to R) \to X$ can be thought of as a *policy function*, that is, 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 (that is, $X \to R$). 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 3.2. 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 $\exists_S : (X \to R) \to R$.

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

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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

For selection functions of existential quantifiers, this construction occurs in 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 the preservation of attainability, we define a product of selection functions, which we show to correspond to the product of its associated quantifiers (*cf.* Definition 2.2).

Definition 3.4. Given selection functions $\varepsilon \in JX$ and $\delta \in JY$, we 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 $p : X \times Y \to R$, we get the function $b : X \to Y$ 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 $\overline{\delta}$ that $\overline{\delta}(\lambda y.p(x, y)) = p(x, b(x))$. This says that for any $x \in X$, the predicate $\lambda y.p(x, y)$ attains its $\overline{\delta}$-value at $b(x)$. Now, a as defined above is such that $\overline{\varepsilon}(\lambda x.p(x, b(x))) = p(a, b(a))$. Again, this says that the predicate $\lambda x.p(x, b(x))$ attains its $\overline{\varepsilon}$-value at a. Putting this all together, we have the following lemma.

Lemma 3.5. $\overline{\varepsilon \otimes \delta} = \overline{\varepsilon} \otimes \overline{\delta}$.

Proof. We calculate

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

by simply unfolding the definitions. □

Remark 3.6. The above definition is equivalent to

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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Example 3.7. 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 the following. 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 $p : X \times Y \to \Omega$, 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 $\forall_X \otimes \exists_Y$, so 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 3.5 and straightforward induction, we get the following theorem.

Theorem 3.8. 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 corollary expresses this in terms of attained values, which is useful for the formulation and justification of the applications we have in mind.

Corollary 3.9. 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 $p : \prod_{i=0}^{n-1} X_i \to R$ attains its Φ-value at $\vec{a} = E(p)$ in the sense that

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

Example 3.10. We continue from Example 2.5 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 3.9, for any game $p : \prod_{i=0}^{n-1} X_i \to \Omega$, 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 $\vec{a}$ since 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.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Remark 3.11. 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 $\gamma : X \to KY$, we 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 $p : X \times Y \to R$.

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 $\delta : X \to JY$, we define their dependent product as

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

for $p : X \times Y \to R$, 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 3.5 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 2.6, and suppose that each quantifier ϕ_i has a selection function ε_i. By the definitions of selection function and optimal strategy, we have the following lemma.

Lemma 3.12. The construction

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

where $w_{\vec{x}}$ is defined in 2.6(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 in their own right.

Lemma 3.13.

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 3.6, taking $X = X_k$, $Y = \prod_{i=k+1}^{n-1} X_i$, $\varepsilon = \varepsilon_k$ and $\delta = \bigotimes_{i=k+1}^{n-1}\varepsilon_i$, $a = a_k$. □

Lemma 3.14.

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

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 3.13 and course-of-values induction. By the assumption that ε_i is a selection function for the quantifier ϕ_i and Theorem 3.8, 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 3.13 and 3.14, we get the following theorem.

Theorem 3.15. The optimal-strategy functions next_k constructed in Lemma 3.12 can be calculated as

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

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 $p_k : X_k \to R$ 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 $k, j < n$.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Theorem 3.15(2) says that the optimal move a_k can be computed from the selection function ε_k and the mapping $\lambda x_k.w_{a_0,\dots,a_{k-1},x_k}$ 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 when all types X_i are the same and equal to X. For example, this can be implemented as follows in Haskell (Hutton 2007):

```
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)
```

Here we use lower case letters `r` and `x` for R and X because of Haskell's syntactical requirements. In Haskell, a finite list of length n is written

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

where [] is the empty list. The operator `otimes` computes the binary product of a selection function $\varepsilon : JX_0$ with a selection function $\delta : J(\bigotimes_{i=1}^{n-1} X_i)$, obtaining a selection function in $J(\bigotimes_{i=0}^{n-1} X_i)$, 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 also works for infinite lists of selection functions (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 $0, 1, \dots, n-1$) and n rounds, with player i picking his move at round i from a fixed set X_i. In standard game theory, a play $(x_0, \dots, x_{n-1}) \in \Pi_{i=0}^{n-1} X_i$ (*cf.* Definition 2.6) is also known as a *strategy profile*, and outcome functions $p : \Pi_{i=0}^{n-1} X_i \to \mathbb{R}^n$ are called *payoff functions*, since $p(x_0, \dots, x_{n-1}) = (v_0, \dots, v_{n-1})$

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 $\varepsilon_i : (X_i \to \mathbb{R}^n) \to X_i$ are

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

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. 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 $\text{next}_k(x_0, \ldots, x_{k-1})$ 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 that 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 $\text{fix} : (R \to R) \to R$ is said to be a *fixed-point operator* if $\text{fix}(p)$ is a fixed point of p for every $p : R \to R$, that is,

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

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 endo-function 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, $f : (R \to R) \to R$ 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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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. 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

$$\begin{aligned} \varepsilon_X(p) &:= \mathrm{fix}_X(\pi_X \circ p) \\ \delta_Y(q) &:= \mathrm{fix}_Y(\pi_Y \circ q). \end{aligned}$$

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

The selection functions ε and δ are not fixed-point operators themselves, and neither are the derived quantifiers $\phi = \overline{\varepsilon} \in KX$ and $\gamma = \overline{\delta} \in KY$. But, by Theorem 3.8, this gives $\overline{\varepsilon \otimes \delta} = \phi \otimes \gamma$, and by the fact that $J(X \times Y) = K(X \times Y)$ for $R = X \times Y$ and $\mathrm{fix}_{X\times Y}$ is a fixed-point operator if and only if it is its own selection function, we conclude from Lemma 4.2 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, however, again in view of Theorem 3.8, 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 $r = (s, t) : X \times Y \to X \times Y$, 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 $r : X \times Y \to X \times Y$ is of the form (s, t) with $s : X \times Y \to X$ and $t : X \times Y \to Y$,

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

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.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

By Lemma 4.2 and induction, we have the following theorem.

Theorem 4.3. If X_i for $0 \leqslant i < n$ 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 discussed further 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 again be explained in terms of products of selection functions. The principle we consider is the *infinite pigeon-hole principle*, which says that for any finite set $n = \{0, 1, \ldots, n-1\}$ of colours and any colouring of the natural numbers, some colour occurs infinitely often:

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

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

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

The dialectica interpretation of this is

$$\begin{aligned} \forall n : \mathbb{N}\, \forall f : \mathbb{N} \to n\, \forall \varepsilon : n \to (\mathbb{N}^{\mathbb{N}} \to \mathbb{N})\, \exists k \in n\, \exists p : \mathbb{N} \to \mathbb{N} \\ p(\varepsilon_k(p)) \geqslant \varepsilon_k(p) \wedge f(p(\varepsilon_k(p))) = k, \end{aligned}$$

that is, given n, f and a sequence ε_i, we must find k and p such that

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

Intuitively, the function p is trying to compute a value j that makes the statement true (that is, the selection function for the existential quantifier in $\exists j \geqslant i(f\, j = k)$), whereas the functional ε_k tries to produce a counter-example i given any such p and fixed colour k (that is, the selection function for the universal quantifier in $\forall i(pi \geqslant i \wedge f(pi) = k)$). 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 $\varepsilon_0, \ldots, \varepsilon_{n-1}$, 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, that is, $p_k(\varepsilon_k(p_k)) = k$.

But, again, we can also view the above as an instance of our general notion of a sequential game (Definition 2.6). Consider the game where $X_i = R = \mathbb{N}$, and $\max : \mathbb{N}^n \to \mathbb{N}$ is the outcome function, and $\phi_k(p) = p(\varepsilon_k p)$ is the outcome quantifier for round k. We will show how the above computational interpretation of the infinite pigeonhole principle

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

can be realised 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 4.4. 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 $p_k(x_k) \geqslant x_k$, and for some $k < n$ we have

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

where

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

In fact, by Theorem 3.15, $x_k = \varepsilon_k(p_k)$, for all $k < n$. Hence

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

and $p_k(x_k) \geqslant x_k$, 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 3.15.

Remark 4.5. 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 *backtracking*. For instance, if each $\varepsilon_k : (\mathbb{B} \to \mathbb{B}) \to \mathbb{B}$ is a selection function for the boolean existential quantifier, and $p(x_0, \dots, x_{n-1})$ is a decidable predicate on n-boolean variables, then

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

computes an assignment that 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 $d : X \times X \to R$ 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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 $d(x_0, x_1) + \cdots + d(x_{k-2}, x_{k-1})$. 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, we 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 4.6. 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 just looking at a few positions, thereby speeding up the backtracking (*cf.* Escardó (2007)). In fact, the product of selection functions behaves like this in general, including in all of the applications mentioned above.

Note also that, alternatively, we could use the dependent version of the product of selection functions (Remark 3.11) to ensure that the next element added to the path is connected to the previous one, and has not been visited before, which means that only proper paths are considered.

5. Infinite products of selection functions

Escardó (2008, Definition 4.5) constructed a functional

$$\Pi : ((D \to \mathscr{B}) \to D)^\omega \to ((D^\omega \to \mathscr{B}) \to D^\omega)$$

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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Escardó (2008, Theorem 4.6) proved that if we are given $\varepsilon \in (JD)^\omega$ such that $\varepsilon_i \in JD$ is a selection function for an existential quantifier $\exists_{S_i}$, with $S_i \subseteq D$, then $\Pi(\varepsilon) \in JD^\omega$ 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 will now rework the product functional Π in a number of ways:

(1) We will work with an infinite sequence X_i of spaces rather than a single domain D, and 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

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

(2) We will allow R to be any discrete space, not just the booleans. In the infinite case, the assumption that R be discrete is essential (Remark 5.11).

(3) We will show that, more generally, if we are given $\varepsilon \in \prod_i JX_i$ such that $\varepsilon_i \in JX_i$ is a selection function for a quantifier $\phi_i \in KX_i$, then $\bigotimes_i \varepsilon_i \in J\left(\prod_i X_i\right)$ is a selection function for a suitably defined quantifier $\bigotimes_i \phi_i \in K\left(\prod_i X_i\right)$.

(4) We will note 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

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

so the infinite version can be seen as simply the iteration of the binary version of the product of selection functions. We will 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 will 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 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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

be characterised in a number of ways, including:

(i) the sequential T_0 spaces with countable pseudo-bases;
(ii) the sequential T_0 spaces with admissible quotient representations;
(iii) a certain full subcategory of the category PER(ωAlgLat), whose objects are the countably based algebraic lattices with a partial equivalence relation, and whose morphisms are the Scott continuous maps that preserve the equivalence relation (Bauer 2002).

QCB spaces admit a theory of computability, and, as shown in 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 account simultaneously for both 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 $n \geqslant 0$, we define

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

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

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

We use n_α to denote 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 it 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 thus eventually

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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, it is seen as a singleton tree consisting of just a leaf, otherwise the subtree of p that follows the branch $x_n \in X_n$ is that corresponding to the predicate $p_{x_n} : \prod_i X_{i+n+1} \to R$. Intuitively, to evaluate $p(\alpha)$ for any given $\alpha \in \prod_i X_{i+n}$, we follow 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 lemma is a counterpart of induction on well-founded trees, and is well known in various guises and particular situations.

Lemma 5.1 (Bar induction). Let X_i be a sequence of k-spaces and R be discrete. Consider a sequence of sets

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

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

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

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

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

We now consider definitions of continuous functionals $h_n : P_n \to Y_n$ by bar recursion (Normann 1999, Section 6), where the spaces P_n are as in Lemma 5.1 and the spaces Y_n are arbitrary. Given $L_n : R \to Y_n$ and $B_n : P_n \times (X_n \to Y_{n+1}) \to Y_n$, 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, it is easy to see 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, it is easy to see that the following lemma holds.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Lemma 5.2. Let $B_n : P_n \times (X_n \to Y_{n+1}) \to Y_n$ 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 $h_n : P_n \to Y_n$ 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 5.3. 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 5.2 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 $\Pi : (JD)^\omega \to JD^\omega$ 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, that is, $\varepsilon'_i = \varepsilon_{i+1}$. 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 selection 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 5.4. Equivalently, to define the variation of the binary product, we can consider the isomorphism

$$X_0 \times \prod_i X_{i+1} \to \prod_i X_i$$

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 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, instead we consider the system of equations

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

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

Lemma 5.5. 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, it satisfies

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

Construction. Define $B_n : P_n \times (X_n \to Y_{n+1}) \to Y_n$ 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 $\delta_n : P_n \to Y_n$, and

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

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 $L_n(r) =$

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

$\varepsilon_n(\lambda x.r) * L_{n+1}(r)$. 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 5.2. □

We emphasise that the next construction is not a specification by bar recursion, since a domain cannot be discrete, except in the uninteresting case that it is the one-point space, and hence the two equations do not uniquely characterise δ_n. But if R is, for example, a lifted discrete space, this, of course, comes very close to a specification by bar recursion, which is what the proof of Theorem 5.7 exploits.

Lemma 5.6. 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.

Proof. Let $F = \prod_n (P_n \to Y_n) = \prod_n J(Y_n)$, and define $H : F \to F$ 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 $\delta_n : P_n \to Y_n$, because F is a domain. Then $\delta_n(p) = B_n(p, \lambda x.\delta_{n+1}(p_x))$ holds by construction. Moreover, it is clear that B_n depends continuously on ε_n, and hence so do H and its least fixed point δ. By induction on k,

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

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

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

Theorem 5.7. 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 PER(ωAlgLat) 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 $|\varepsilon_i| : (|X_i| \to |R|) \to |X_i|$ that preserve $\sim$, since the embedding preserves function spaces. Then we can apply Lemma 5.6 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 5.6 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 $s_o, s_1 : \prod_n |X_{i+n}| \to |R|$ track p, then $|\delta_n|(s_0) \sim |\delta_n|(s_1)$. Hence $s_0 \sim s_1$ implies

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

$|\delta_n|(s_0) \sim |\delta_n|(s_1)$, so $|\delta_n|$ is a morphism of PER(ωAlgLat), which then gives a morphism δ_n of QCB, because the embedding of QCB into PER(ωAlgLat) is full. □

Note that the assumption of discreteness of R is used twice in this proof, so bar induction can be applied to:

(1) establish that there is at most one solution in Lemma 5.5;

(2) prove totality of the functional constructed in Lemma 5.6.

As discussed in Remark 5.11 below, such an assumption is essential.

Lemma 5.8. Under the assumptions and notation of Theorem 5.7, 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 5.7 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. □

Corollary 5.9. The equation $F_n(\varepsilon) = \varepsilon_0 \otimes F_{n+1}(\varepsilon')$ has a unique continuous solution $F_n : \prod_i JX_{i+n} \to J\left(\prod_i X_{i+n}\right)$, 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 5.8, and holds uniquely by Theorem 5.7. □

Definition 5.10. For any sequence $\varepsilon_i \in JX_i$, we write $\bigotimes_i \varepsilon_i := \delta_0(\varepsilon)$. Then, by Corollary 5.9, this is characterised as

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

Remark 5.11. 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 5.7. This is adapted from a similar counter-example in Berger and Oliva (2005). Note, however, that for specific families ε_n, a solution may exist for R non-discrete, for example, if the ε_n are constant functions. But notice that, by virtue of the previous counter-example, the equation of Corollary 5.9 cannot have a solution if $R = (\mathbb{N} \to \mathbb{N})$.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

5.6. *Infinite products of quantifiers*

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

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

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, if we 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 the $\prod_i S_i$ 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 $\left(\bigotimes_i \exists_{S_i}\right)(p)$ 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 5.7, 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, for example, $X_i = R = \mathbb{N}$ and $\phi_i(q) = q(0) + 1$, 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.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Now, by the discussion that follows Definition 3.1, if the quantifiers ϕ_n are attainable, $\phi_n(\lambda x.r) = r$, and hence the above constraint on the sequence L_n amounts to $L_n(r) = L_{n+1}(r)$, that is, L_n can be any constant sequence. Then Lemma 5.2 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, the system of equations $\gamma_n = \phi_n \otimes \gamma_{n+1}$ has at most one attainable solution.

Theorem 5.12. 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 5.7, there is a unique sequence δ_i such that $\delta_n = \varepsilon_n \otimes \delta_{n+1}$. Taking $\gamma_n = \overline{\delta_n}$, Theorem 3.8 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. □

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

$$\begin{aligned} \bigotimes_i \phi_i &= \phi_0 \otimes \bigotimes_i \phi_{i+1} \\ \overline{\bigotimes_i \varepsilon_i} &= \bigotimes_i \overline{\varepsilon_i}. \end{aligned}$$

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

$$\begin{aligned} \bigotimes_i \exists_{X_i} &= \exists_{\prod_i X_i} \\ \bigotimes_i \forall_{X_i} &= \forall_{\prod_i X_i} \\ \bigotimes_i \sup_{X_i} &= \sup_{\prod_i X_i} \\ \bigotimes_i \inf_{X_i} &= \inf_{\prod_i X_i}, \end{aligned}$$

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

Question 5.13. By Escardó (2008, Theorem 6.3), if $R = \mathbb{B}$, if X is a space of Kleene–Kreisel functionals, and if $\varnothing \neq S \subseteq X$ 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. Note 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 what 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.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

5.7. *Relation to traditional instances of bar recursion*

Recall that in Lemma 3.13 we developed a recursive characterisation of the optimal play

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

Before we proceed, we shall 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 that 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 5.14. For $\varepsilon_i \in JX_i$ and $p : \prod_{i=0}^{n-1} X_i \to R$ fixed, and for each $k < n$, define

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

by

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

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 5.15. The family of functions $\mathrm{cbr}_k : \prod_{i=0}^{k-1} X_i \to \prod_{i=0}^{n-1} X_i$ 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

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

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

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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

This functional cbr is very similar to (but 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,

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

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

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

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,

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

6. The continuation and selection monads

Crucial parts of the above development follow naturally from conceptual arguments 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 J is also part of a monad, which we refer to as the *selection monad*. The two monads are strong, which explains the products of quantifiers and the products of selection functions in a unified way. Moreover, the procedure $\varepsilon \mapsto \overline{\varepsilon}$ that transforms selections functions into quantifiers given in Definition 3.3 is a monad morphism $J \to K$. This explains our main Theorem 3.8, which 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 $\mathscr{X}$ is a triple (T, η, μ) where $T : \mathscr{X} \to \mathscr{X}$ is a functor, and $\eta_X : X \to TX$ (the unit) and $\mu_X : TTX \to TX$ (the multiplication) are natural transformations, subject to the three equations

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

It is fairly laborious and space consuming to check the associativity law 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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 $\mathscr{X}$-enriched (Kock 1970a). This means that its action on morphisms is tracked by a morphism

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

of $\mathscr{X}$, rather than just a function from the hom-set $\mathscr{X}(X, Y)$ to the hom-set $\mathscr{X}(TX, TY)$. 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 $\mathscr{X}$-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 equations referred to. Notice that $\lambda y.(x, y) : Y \to X \times Y$, so $T(\lambda y.(x, y)) : TY \to T(X \times Y)$.

Definition 6.1. Let T be a strong monad on a cartesian closed category $\mathscr{X}$. We define a morphism

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

by

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

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

Remarks 6.2.

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 $\theta_X : TX \to T'X$ commutes with the standard monoidal monad structure of Definition 6.1:

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

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.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

6.2. *The continuation monad*

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

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

This is an $\mathscr{X}$-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 $\mathscr{X}$. This functor is self-adjoint on the right,

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

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

For a morphism $f : X \to Y$, the morphism $Kf : KX \to KY$ 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 $t_{X,Y} : X \times KY \to K(X \times Y)$ 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 $\eta_X : X \to KX$ is defined by

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

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

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

Remark 6.3. It is easy to verify that the morphism $m_{X,Y} : KX \times KY \to K(X \times Y)$ defined in Definition 6.1 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 6.2, 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 the current paper.

Examples 6.4. Consider $R = \Omega$ in the category of sets or any topos.

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

$$\begin{aligned} Kf(\exists_A) &= \exists_{f(A)} \\ Kf(\forall_A) &= \forall_{f(A)}, \end{aligned}$$

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

since

$$\exists y \in f(A)(q(y)) = \exists x \in A(q(f(x))),$$

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 $t_{X,Y} : X \times KY \to K(X \times Y)$, we have, for any $x \in X$ and any $B \subseteq Y$,

$$\begin{aligned} t(x, \exists_B) &= \exists_{\{x\} \times B} \\ t(x, \forall_B) &= \forall_{\{x\} \times B}. \end{aligned}$$

Similarly, by Example 2.3, for the operation $m_{X,Y} : KX \times KY \to K(X \times Y)$, we have for any $A \subseteq X$,

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

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

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

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

4 The multiplication $\mu_X : KKX \to KX$ 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 $B_\phi \subseteq X$, that is,

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

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 $\mathscr{X}$. In order to define this manifestation of the Kleisli category of J, we will work simultaneously with a manifestation of the Kleisli category of K.

We have used letters X, Y, Z for the objects of our underlying category $\mathscr{X}$. In order both to avoid confusion and to be compatible with the notational conventions used in 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 $\mathscr{X}$ 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 $\mathscr{X}$, which, by transposition, amounts to a morphism

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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

$$P : \mathscr{X} \to \mathscr{X}^{\mathrm{op}},$$

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 $P : \mathscr{X} \to \mathscr{X}^{\mathrm{op}}$ be an enriched functor that is self-adjoint on the right in the sense that there is a natural isomorphism

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

Note 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 define new morphisms and enriched functors using the lambda-calculus.

Definition 6.5. Define a new category $\mathscr{K}$ from our underlying category $\mathscr{X}$ and the functor $P : \mathscr{X} \to \mathscr{X}^{\mathrm{op}}$ as follows:

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

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

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

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

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

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

Lemma 6.6. The functor $F = F_K : \mathscr{X} \to \mathscr{K}$ that is the identity on objects and sends $f : X \to Y$ to $Pf : PY \to PX$, regarded as a morphism $X \to Y$ of $\mathscr{K}$, has a right adjoint $G = G_K : \mathscr{K} \to \mathscr{X}$.

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

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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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 6.7. We define a new category $\mathscr{J}$ from our underlying category $\mathscr{X}$ and the enriched functor $P : \mathscr{X} \to \mathscr{X}^{\mathrm{op}}$ as follows:

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

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

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

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

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

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

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

$$f : PB \to (A \to B)$$

$$g : PC \to (B \to C)$$

in $\mathscr{X}$, we define the composite

$$g \,\square\, f : PC \to (A \to C)$$

by, for any $r : PC$,

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

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

Claim.

$$H(g \,\square\, f) = Hf \circ Hg.$$

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Proof of the claim. We have

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

□

Proof of associativity. Let

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

and $s : PD$, and calculate:

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

□

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

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

Lemma 6.8. The functor $F = F_J : \mathscr{X} \to \mathscr{J}$ that is the identity on objects and that sends a morphism $f : X \to Y$ to the constant f morphism $PY \to (X \to Y)$ of $\mathscr{X}$ has a right adjoint $G = G_J : \mathscr{J} \to \mathscr{X}$.

Proof. We describe G and the required natural isomorphism

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

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

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

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

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

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

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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

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

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

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 $\epsilon : FGA \to A$ be a natural transformation in A, as follows. Given a morphism $h : A \to B$ in $\mathscr{J}$, the morphism Gh in $\mathscr{X}$ is the unique $g : GA \to GB$ such that $\epsilon \,\square\, Fg = f$, where $f = h \,\square\, \epsilon : FGA \to B$. Now, by the construction of g from f in the proof of Lemma 6.8, expanding all the definitions and using the fact that a morphism $h : A \to B$ of $\mathscr{J}$ is a morphism $h : PB \to (A \to B)$ of $\mathscr{X}$, 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 6.8 does not say explicitly what the unit $\eta_X : X \to GFX$ 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 $g = \mathrm{id}_{FX} : FX \to FX$. By the construction of g from f in the proof of Lemma 6.8 and the definition of the identities of $\mathscr{J}$,

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

Applying the standard construction of a monad from a given adjunction, we get the following lemma.

Lemma 6.9. $JX = GFX = (PX \to X)$ is a strong monad on $\mathscr{X}$, with its action on morphisms $f : X \to Y$ given by, for all $\varepsilon : JX$ and $q : PY$,

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

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

$$\begin{aligned} \eta_X(x)(p) &= x \\ \mu_X(E)(p) &= E(P\epsilon_p(p))(p). \end{aligned}$$

Proof. Let $f : X \to Y$ in $\mathscr{X}$ and $h = Ff$. We have $h_q = f$ by the definition of F, so we can conclude that

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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

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

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

Lemma 6.10. There is a monad morphism $\theta_X : JX \to KX$, given by the adjoint transposes of the family of maps

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

Proof. We apply Moggi (1990, Proposition 4.0.10), which shows that monad morphisms $\theta : J \to K$ are in bijection with functors $H : \mathscr{X}_J \to \mathscr{X}_K$ of the Kleisli categories that 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 $\theta_X : JX \to KX$ of the natural transformation as Hh, where the morphism h in $\mathscr{X}_J$ is the identity $JX \to JX$ of X in $\mathscr{X}$, regarded as a morphism $JX \to X$ of the Kleisli category $\mathscr{X}_J$ of J. In the manifestation $\mathscr{J}$ of $\mathscr{X}_J$, this amounts to a morphism $h : PX \to (JX \to X)$ of $\mathscr{X}$, which is readily seen to be $h_p(\varepsilon) = \varepsilon(p)$. Note that $Hh : JX \to X$, because H is the identity on objects, so $Hh : JX \to KX$ regarded as a morphism of the Kleisli category of K, and hence $Hh : PX \to PJX$ regarded as a morphism of $\mathscr{X}$. Now, for $H : \mathscr{J} \to \mathscr{K}$ constructed as in Definition 6.7, 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. □

Theorem 6.11. $JX = ((X \to R) \to X)$ is a strong monad on $\mathscr{X}$, with action on morphisms $f : X \to Y$ given by

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

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

$$\begin{aligned} \eta_X(x)(p) &= x \\ \mu_X(E)(p) &= E(\lambda\varepsilon.p(\varepsilon(p)))(p). \end{aligned}$$

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 $\lambda\varepsilon.\lambda p.p(\varepsilon(p))$, so we get the desired monad morphism. □

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Remarks 6.12.

1 Because a monad morphism commutes with the functors that define the monads, Theorem 6.11 gives, for any $f : X \to Y$ and $\varepsilon \in JX$:

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

Hence, if $\varepsilon \in JX$ is a selection function for the quantifier $\phi \in KX$, we have that $Jf(\varepsilon)$ is a selection function for the image quantifier $Kf(\phi)$. In particular, by Example 6.4(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 3.8 follows directly from Theorem 6.11 and Remarks 6.2.

3 The construction of the strength and of the monoidal monad structure given in Definition 6.1 is characterised as follows, where $\otimes$ is defined as in Section 3.1:

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

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

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

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

(c) Hence, by Remark 6.2, 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 lays the foundations for applications to proof theory that we are currently developing. We are studying 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 realiser/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 that 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 $\bigotimes$ realises (in the sense of Kreisel's modified realisability) the J-shift

$$\forall n(JA(n)) \to J\,(\forall n A(n))\,.$$

The J-shift is more general than the double negation shift (K-shift), and gives the K-shift in the cases it exists, like 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 that realises the axiom of countable (and dependent) choice.

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

Acknowledgements

We would like to thank Matías Menni for discussions on the subject of Section 6. We also thank the anonymous referees for their careful reading of the paper and for helpful suggestions.

References

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

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

Battenfeld, I., Schröder, M. and Simpson, A. (2006) Compactly generated domain theory. *Mathematical Structures in Computer Science* **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. *Electronic Notes in Theoretical Computer Science* **172** 69–99.

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 – H. Bekič (1936–1982) Selected Papers edited by C. B. Jones. *Springer-Verlag Lecture Notes in Computer Science* **177**.

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. In: Baaz, M., Friedman, S. D. and Kraijcek, J. (eds.) Logic Colloquium '01. *Springer-Verlag Lecture Notes in Logic* **20** 89–107.

Berger, U. and Oliva, P. (2006) Modified bar recursion. *Mathematical Structures in Computer Science* **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*, IEEE Computer Society 443–452.

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

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.

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-Verlag.

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press

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. *Springer-Verlag Lecture Notes in Mathematics* **811**.

Normann, D. (1999) The continuous functionals. In: Griffor, E. R. (ed.) *Handbook of Computability Theory*, Chapter 8, North-Holland 251–275.

Oliva, P. (2006) Understanding and using Spector's bar recursive interpretation of classical analysis. In: Beckmann, A., Berger, U., Löwe, B. and Tucker, J. V. (eds.) Logical approaches to computational barriers. Proceedings second conference on computability in Europe, CiE 2006, Swansea. *Springer-Verlag Lecture Notes in Computer Science* **3988** 423–434.

Schröder, M. (2002) Extended admissibility. *Theoretical Computer Science* **284** (2) 519–538.

Smyth, M. (1977) Effectively given domains. *Theoretical Computer Science* **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. (ed.) *Recursive Function Theory: Proc. Symposia in Pure Mathematics* **5**, American Mathematical Society 1–27.

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

https://doi.org/10.1017/S0960129509990351 Published online by Cambridge University Press