A categorical setting for lower complexity

Robin Cockett[1],[3], Brian F. Redmond[2],[4]

Department of Computer Science
University of Calgary
Calgary, Canada

Abstract

A polarized strong category consists of a cartesian category, $\mathbb{X}$, and a category $\mathbb{Y}$, together with a module $M : \mathbb{X} \times \mathbb{Y} \longrightarrow \mathbb{Y}$ equipped with a strong composition and identities. These categories can be used to provide an abstract setting for investigating computational setting with complexity below primitive recursive. This paper develops the theory of polarized strong categories, explains how they relate to the theory of fibrations, and provides a concrete example which illustrates their applicability to these lower complexity systems of computation.

Keywords: Polarized categories, fibrations, recursion principles, complexity.

1 Introduction

It is well-known that having a (strong) natural number object in a monoidal category immediately delivers all primitive recursive functions [19]. Therefore, to obtain settings which realize lower complexities something quite drastic has to be done.

An attractive feature, however, of a natural number object is that, as an initial data type, it arrives packaged with a universal property which enforces certain basic equalities on the maps involving that type. In the initial models of a doctrine involving such a type, these equalities become the basis for generating *all* the equality judgements. As computation is often viewed as

---

[3] Email: robin@ucalgary.ca

[4] Email: bredmond@ucalgary.ca

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

arising from initial settings, this native notion of equality is of considerable interest.

In dealing with settings which are below primitive recursive our interest is not merely in the presence or absence of maps but also in the notion of equality which they support. Therefore, we would like data types, even in these settings, to deliver a universal property and whence a native notion of equality.

This paper describes for lower complexity settings how initial inductive data types – together with their corresponding universal property – can be supplied by moving to polarized strong categories. We illustrate this with a simple semantic setting, which we develop in some detail, which uses $\mathcal{R}$-sized sets[5] and polynomially bounded maps. The significance of this example is that its arithmetic power is strictly less than primitive recursive (because of the explicit polynomial bound) although it is at least as powerful as PSPACE (as only the size of the output is controlled).

The example, of course, opens the question of whether the full system we describe can capture Polynomial Time (PTIME) and Polynomial Space (PSPACE) computations. For discussion of these matters see the conclusion of this paper and the report [20] in which a much fuller type theory is discussed in support of a PTIME programming language called **Pola** which is currently being developed by Mike Burrell [3].

The focus of the current paper is on the categorical doctrines[6] which can be used for modeling the proof theories of these lower complexity settings.

The nucleus from which this paper grew was the realization that the system of Bellantoni and Cook [2] for describing PTIME had an immediate interpretation as a proof theory for a polarized logic. Polarities were introduced by Girard [6] to classify the behavior of the logical connectives in his "constructive" classical logic LC – his idea was directly related to Andreoli's notion of *focussing* [1]. Olivier Laurent [14] further developed these ideas and quickly realized that there was a compelling connection to games [15]: these and further references to related developments are described in [7].

The general categorical proof theory for polarized logics and games is described in [4] and uses the notion of a polarized category. A polarized category is simply a module[7], however, viewed as a categorical structure in its own right. Polarization is produced by the separation beween the category which

---

[5] These play a similar role to the "length spaces" of [9].

[6] The program of expressing complexity classes categorically is not new. Notable was Jim Otto's [18] pioneering work. The first author would particularly like to acknowledge his conversations with Jim – they were formative in this work.

[7] A module $M : \mathbb{X} \longrightarrow \mathbb{Y}$ is variously called a profunctor, a distributor, a *bi*module: it is equivalently a functor $M : \mathbb{X}^{\mathrm{op}} \times \mathbb{Y} \longrightarrow \mathbf{Set}$ or a "bipartite" category consisting of the categories $\mathbb{X}$ and $\mathbb{Y}$ and in addition "cross maps" running from the objects of $\mathbb{X}$ to objects of $\mathbb{Y}$ – but not in the reverse direction.

is the domain of the module (the "opponent" world) and the category which is the codomain (the "player world"). While Bellantoni and Cook used the terms "ordinary" and "safe" (respectively) for the two worlds of computation – rather than the game theory inspired terminology used here – it was clear that they were employing the technique of polarizing to achieve separation for computation.

Bellantoni and Cook's system of safe recursion, which only considered binary natural numbers, was a simplification of a slightly earlier system developed by Leivant [16] which had general inductive data types and infinitely many tiers (although two sufficed). Both these systems allowed duplication in their "safe" worlds and focused on controlling the recursion. The categorical doctrine we present, however, uses a further crucial idea introduced by Hofmann [10]: he realized that it was advantageous to assume that the player (or safe) world is affine – so one cannot duplicate data. This allows one to restrict the "safe" or player world to constant time computations (in context). The intuition is then that "iterating" (parameterized) constant time computations keeps one within polynomial time.

Of course, a categorical doctrine which demands that the player world be affine is quite happy to accept as a model a system which has duplication as well. Thus, the above mentioned systems are not ruled out from being a model of a categorical doctrine which assumes an affine player world.

However, Hofmann's reason for insisting on an affine world was more fundamental: he had observed that one could not allow certain reasonable patterns of recursion over trees – such as counting their leaves (see section 5.3) – at the same time as allowing their duplication. This is because, with duplication, one can easily construct an exponential size tree by simply repeatedly adding a root node whose children are duplicates of the tree constructed so far. Counting the leaves of such a tree takes one outside PTIME. Leivant's system evaded this problem by supporting a recursion principle which did not permit one to count the leaves of a tree.

There is, thus, a trade-off between the power of the recursion principle and allowing duplication in the player world. In the main system presented here, the player world is higher-order and this commits us to a powerful recursion principle and, whence, to limiting duplication. Of course, we could equally well have taken the other approach: allowing duplication and limiting the recursion principle. However, the power of the recursion principle significantly impacts expressiveness and this severely limits the utility of systems with restrictive recursion principles. Thus, here we have chosen the other direction, namely to promote the use of a more powerful recursion principle, as we feel it holds more promise.

3

Acknowledgements

We thank Geoff Cruttwell for reading an early draft of the paper and for making many useful suggestions and observations.

2 The basic categorical setting

The basic categorical framework of this paper consists of a cartesian category $\mathbb{X}$, a category $\mathbb{Y}$, and a module connecting $\mathbb{X} \times \mathbb{Y} \longrightarrow \mathbb{Y}$ creating a polarized category [4] which is, in addition *strong*. The category $\mathbb{X}$ will referred to as the **opponent category** (or opponent world) and the category $\mathbb{Y}$ is referred to as the **player category** (or player world) while the module maps are referred to as **cross maps**.

Polarized strong categories are closely related to certain fibrations (over the opponent world). This provides an alternative and compelling perspective on these setting which we shall wish to exploit. This section, therefore, develops the relationship to fibrations and also introduces $\mathcal{R}$-sized sets which we shall use as a running example to illustrate the theory.

2.1 Polarized strong categories

A **polarized strong category** consists of a cartesian category[8] $\mathbb{X}$ (the opponent world), and a category $\mathbb{Y}$ (the player world), and a module M:

$$M : \mathbb{X} \times \mathbb{Y} \longrightarrow \mathbb{Y}$$

equipped with a "strong" composition and "strong" identities for the module maps:

$$\frac{(X_1, Y_1) \xrightarrow{f} Y_2 \quad (X_2, Y_2) \xrightarrow{g} Y_3}{(X_1 \times X_2, Y_1) \xrightarrow{f;g} Y_3} \qquad \overline{(\mathbf{1}, Y) \xrightarrow{i_Y} Y}$$

which satisfy:

- Strong identities are natural: $i_Y y = (1, y)i_{Y'}$, for any $y : Y \longrightarrow Y'$ in $\mathbb{Y}$;

- The strong composition preserves the basic module structure: $(f; f')y = f; (f'y)$, $(x_1 \times x_2, y)(f_1; f_2) = ((x_1, y)f_1); (x_2, 1)f_2$, and $(1 \times x, 1)((fy); f') = f; (x, y)f'$;

- $(\pi_1, 1)f = i_Y; f : (\mathbf{1} \times X, Y) \longrightarrow Y'$ and $(\pi_0, 1)f = f; i_{Y'} : (X \times \mathbf{1}, Y) \longrightarrow Y'$;

- $(a_\times, 1)((f_1; f_2); f_3) = f_1; (f_2; f_3) : (X_1 \times (X_2 \times X_3), Y) \longrightarrow Y'$.

In order to demonstate the connection to fibrations we shall want to consider a fixed opponent world and a varying player world and to facilitate this

---

[8] Here this means having finite products. In fact, having a tensor also suffices for the basic strong composition structure; see [23].

we shall refer to a polarized strong category with opponent world $\mathbb{X}$ as an **$\mathbb{X}$-strong category**. An **$\mathbb{X}$-strong functor** $F : \mathbb{Y} \rightarrow \mathbb{Y}'$ between $\mathbb{X}$-strong categories will then be an ordinary functor $F : \mathbb{Y} \rightarrow \mathbb{Y}'$ and a morphism, also labeled F, on cross maps such that:

$$\frac{(X, Y) \xrightarrow{f} Y'}{(X, F(Y)) \xrightarrow{F(f)} F(Y')}$$

which preserve the basic module structure $(x, F(y))F(h)F(y') = F((x, y)hy')$ and preserves the strong composition and identities:

- $F(i_Y) = i_{F(Y)}$
- $F(f_1; f_2) = F(f_1); F(f_2)$

Clearly the composite of two $\mathbb{X}$-strong functors is again an $\mathbb{X}$-strong functor. An **$\mathbb{X}$-strong transformation** between strong functors is an ordinary transformation between the ordinary functors $\alpha : F \rightarrow F'$ such that for cross maps h we have $(1, \alpha)F'(h) = F(h)\alpha$. We now have:

Proposition 2.1 $\mathbb{X}$-*strong categories, functors, and transformations form a 2-category, written* $\mathsf{Str}(\mathbb{X})$.

2.2 Fibrational interpretation

Given an $\mathbb{X}$-strong category $\mathbb{Y}$ a fibration $p : \widetilde{\mathbb{Y}} \rightarrow \mathbb{X}$ can be constructed [9], called the **bundle fibration** $\mathsf{bun}(\mathbb{Y})$, where the total category $\widetilde{\mathbb{Y}}$ of this fibration is defined as follows:

Objects: pairs of objects $(X, Y) \in \mathbb{X} \times \mathbb{Y}$;

Maps: A map from (X_1, Y_1) to (X_2, Y_2) is a pair (x, h), where $x : X_1 \rightarrow X_2$ in $\mathbb{X}$ and $h : (X_1, Y_1) \rightarrow Y_2$ is a module map;

Composition: let $(x, h) : (X_1, Y_1) \rightarrow (X_2, Y_2)$ and $(x', h') : (X_2, Y_2) \rightarrow (X_3, Y_3)$; then composition is defined by $(xx', (\Delta, 1)(h; (x, 1)h')) : (X_1, Y_1) \rightarrow (X_3, Y_3)$;

Identities: $(1_X, (!_X, 1)i_Y) : (X, Y) \rightarrow (X, Y)$.

It is not hard to check that $\widetilde{\mathbb{Y}}$ is a category and moreover gives rise to a fibration:

Proposition 2.2 *The* $\mathsf{bun}(\mathbb{Y})$ *given by the projection functor* $p : \widetilde{\mathbb{Y}} \rightarrow \mathbb{X}$ *defined by* $p(X, Y) = X$ *and* $p(x, h) = x$ *is a fibration over* $\mathbb{X}$ *with a cleavage.*

---

[9] This is not the usual Grothendieck fibration from the module but uses the extra composition ';' of an $\mathbb{X}$-strong category in an essential way.

Proof. For each map $x : X \longrightarrow X'$ and object (X', Y') over X', the cartesian lifting is defined as $f = (x, (!_X, 1)i_{Y'}) : (X, Y') \longrightarrow (X', Y')$. Indeed, f is cartesian over x as $p(f) = x$ and given $g = (z, h) : (Z_1, Z_2) \longrightarrow (X', Y')$ such that z factors as $x'x$, there exists a bundle map $m = (x', h) : (Z_1, Z_2) \longrightarrow (X, Y')$ such that $p(m) = x'$ and:

$$
\begin{aligned}
mf &= (x', h)(x, (!_X, 1)i_{Y'}) \\
&= (x'x, (\Delta, 1)(h; (x', 1)(!_X, 1)i_{Y'})) \\
&= (z, (\Delta, 1)(h; (!_{Z_1}, 1)i_{Y'})) \\
&= (z, (\Delta, 1)(1 \times !_{Z_1}, 1)(h; i_{Y'})) \\
&= (z, (\Delta, 1)(1 \times !_{Z_1}, 1)(\pi_0, 1)h) \\
&= (z, h) = g
\end{aligned}
$$

Moreover, if $m' = (x'', h')$ also satisfies these conditions, then $x'' = p(m') = x'$ and $m'f = (z, h') = g = (z, h)$, so $h = h'$ and m is unique. $\qquad\square$

Proposition 2.3 *This assignment extends to a 2-functor* $\mathsf{bun} : \mathsf{Str}(\mathbb{X}) \longrightarrow \mathsf{CFib}(\mathbb{X})$ *which, moreover, preserves products.*

Proof. The preservation of products is immediate from the construction.

An $\mathbb{X}$-strong functor F extends to a cartesian functor $\widetilde{F}$ on the fibration as follows:

$$
\begin{aligned}
\widetilde{F}(X, Y) &= (X, F(Y)) \\
\widetilde{F}(x, h) &= (x, F(h))
\end{aligned}
$$

This preserves the cleavage as $\widetilde{F}(x, (!_X, 1)i_Y) = (x, F((!_X, 1)i_Y)) = (x, (!_X, 1)i_{F(Y)})$ and is a functor as:

$$
\begin{aligned}
\widetilde{F}(1_X, i_Y) &= (1_X, F((!_X, 1_Y)i_Y)) \\
&= (1_X, (!_X, F(1_Y))F(i_Y)) \\
&= (1_X, (!_X, 1_{F(Y)})i_{F(Y)})
\end{aligned}
$$

and:

$$
\begin{aligned}
\widetilde{F}(x, h); \widetilde{F}(x', h') &= (x, F(h))(x', F(h')) \\
&= (xx', (\Delta, 1)(F(h); (x, 1)F(h'))) \\
&= (xx', (\Delta, 1)(F(h); F((x, 1)h'))) \\
&= (xx', (\Delta, 1)(F(h; (x, 1)h'))) \\
&= (xx', F((\Delta, 1)(h; (x, 1)h'))) \\
&= \widetilde{F}((x, h)(x', h'))
\end{aligned}
$$

The cartesian lifting of a map $x : X \longrightarrow X'$ in $\mathbb{X}$ is $(x, (!_X, 1)i_Y) : (X, Y) \longrightarrow (X', Y)$. This map is preserved as $\widetilde{F}((x, (!_X, 1_Y)i_Y)) = (x, F((!_X, 1_Y)i_Y)) = (x, (!_X, 1_{F(Y)})i_{F(Y)})$. Therefore $\widetilde{F}$ is a cartesian functor.

A $\mathbb{X}$-strong natural transformation $\alpha : F \longrightarrow G$ extends to a cartesian natural transformation $\widetilde{\alpha} : \widetilde{F} \longrightarrow \widetilde{G}$ as follows: the components of the natural transformation are $(1_X, (!_X, 1)\alpha_Y) : (X, F(Y)) \longrightarrow (X, G(Y))$. It is easy to check that this in fact defines a cartesian natural transformation. $\square$

Given any cartesian category $\mathbb{X}$ then $\mathbb{X}$, itself, can be regarded as a $\mathbb{X}$-strong category by letting the cross maps be:

$$\frac{X_1 \times X \xrightarrow{h} X_2}{(X, X_1) \xrightarrow{h} X_2}$$

This makes these cross maps "maps in context". Note that this makes $f; g = a_\times(f \times 1)g$. A $\mathbb{X}$-strong functor F then becomes a strong functor in the usual sense [13,5,23].

We now show that there is a functor in the reverse direction: that is, given any fibration (here we consider fibrations with a cleavage) over a cartesian category $\mathbb{X}$ the fiber over $\mathbf{1}$ naturally forms an $\mathbb{X}$-strong category.

Proposition 2.4 *There is a 2-functor* $\mathsf{pol} : \mathsf{CFib}(\mathbb{X}) \longrightarrow \mathsf{Str}(\mathbb{X})$.

Proof. (Sketch) Let $p : \mathbb{Y} \longrightarrow \mathbb{X}$ be a fibration with cleavage. Then we may build an $\mathbb{X}$-strong category on the fiber over $\mathbf{1}$, $p^{-1}(\mathbf{1})$, where one defines a cross map of the form $(X, Y) \longrightarrow Y'$ as a map $!_X^*(Y) \longrightarrow Y'$ in $\mathbb{Y}$. The strong identity maps are the identity maps in $p^{-1}(\mathbf{1})$. The strong composition of $(X_1, Y) =!_{X_1}^*(Y) \xrightarrow{f} Y'$ and $(X_2, Y') =!_{X_2}^*(Y') \xrightarrow{g} Y''$ is given by lifting the first map to the map $(X_1 \times X_2, Y) \xrightarrow{\widetilde{f}} (X_2, Y')$ (as illustrated below) and

composing with g:

A morphism of fibrations $F : (p : \mathbb{Y} \to \mathbb{X}) \to (q : \mathbb{Y}' \to \mathbb{X})$ consists of a functor $F : \mathbb{Y} \to \mathbb{Y}'$ such that $p = F; q$ which preserves the cleavage. The restriction to the fiber over $\mathbf{1}$ then defines an $\mathbb{X}$ strong functor between the induced $\mathbb{X}$-strong categories. Similarly a natural transformation between a morphism of fibrations induces an $\mathbb{X}$-strong transformation. $\square$

Proposition 2.5 *The above functors forms an adjunction* $\mathsf{bun} \dashv \mathsf{pol} : \mathsf{Str}(\mathbb{X}) \to \mathsf{CFib}(\mathbb{X})$.

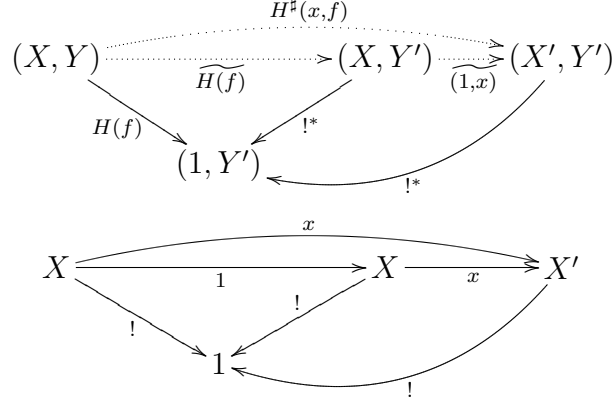**Proof.** (Sketch) The unit of the adjunction carries $\mathbb{Y}$ to the fiber over $\mathbf{1}$ in $\mathsf{bun}(\mathbb{Y})$:

$$\eta : \mathbb{Y} \to \mathsf{pol}(\mathsf{bun}(\mathbb{Y})); \quad \begin{array}{c} Y \\ \downarrow f \\ Y' \end{array} \mapsto \begin{array}{c} (\mathbf{1}, Y) \\ \downarrow (1_{\mathbf{1}}, i_Y f) \\ (\mathbf{1}, Y') \end{array}$$

We need to show the following universal property:

$$\begin{array}{ccc} \mathbb{Y} & \xrightarrow{\eta} & \mathsf{pol}(\mathsf{bun}(\mathbb{Y})) \\ & \searrow{\scriptstyle H} & \downarrow{\scriptstyle \mathsf{pol}(H^\sharp)} \\ & & \mathsf{pol}(\mathbb{A}) \end{array}$$

To do this we indicate how $H^\sharp$ is defined on $(x, f) \in \mathsf{bun}(\mathbb{Y})$ using the lifting property (dotted maps) of the fibration $\mathbb{A}$ and the definition $(X, A) := !^*_X(Y) \in$

8

$p^{-1}(X)$ as above:

This is clearly unique as the liftings are unique.

To show this is a Galois adjunction it suffices to check that $\mathsf{bun}(\eta)$ is an isomorphism: however, this morphism of fibration is determined by its effect on the fiber over $\mathbf{1}$ and these fibers are essentially the same category. $\qquad\square$

The fact that there is a Galois adjunction between $\mathsf{Str}(\mathbb{X})$ and $\mathsf{CFib}(\mathbb{X})$ means that one can identify a common full subcategory: the subcatgory of $\mathsf{CFib}(\mathbb{X})$ corresponds to "bundle fibrations" while the full subcategory of $\mathsf{Str}(\mathbb{X})$ corresponds more prosaically to the $\mathbb{X}$-strong categories in which $\mathbb{Y}$ is already the fibre over $\mathbf{1}$ in $\mathsf{bun}(\mathbb{Y})$.

2.3 The category of $\mathcal{R}$-sized-sets

Let $\mathcal{R} = (\mathcal{R}, +, \cdot, 0, 1, \leq)$ be a size rig: that is an ordered (commutative) rig $\mathcal{R}$ with bottom element 0 (that is $0 \leq r$ for all $r \in \mathcal{R}$), and order-preserving operations. The canonical rig we have in mind is the natural numbers $\mathbb{N}$ (but $\mathbb{R}_{\geq 0}$ or $\mathbb{Q}_{\geq 0}$ will also do).

The category of $\mathcal{R}$-sized sets with polynomially bounded maps, denoted $\mathcal{R}\text{-}\mathsf{Set}_{\mathsf{poly}}$, is constructed as follows:

Objects: an object is a function $\alpha : A \longrightarrow \mathcal{R}$. We may think of this as a set A, each of whose elements is assigned an abstract size (i.e. an element of $\mathcal{R}$). These are called $\mathcal{R}$-sized sets;

Maps: A map from $\alpha : A \longrightarrow \mathcal{R}$ to $\beta : B \longrightarrow \mathcal{R}$ is a function $f : A \longrightarrow B$ such that there exists a polynomial $p \in \mathcal{R}[x]$ satisfying: for all $a \in A$, $\beta(f(a)) \leq p(\alpha(a))$. In this case we say that p is a bound for f;

Composition: The usual function composition. The composite is bounded by substitution of polynomials. Identities are the identity functions with identity polynomial bounds.

It is clear that this forms a category. It is furthermore a cartesian category.

The product of $\mathcal{R}$-sized sets $\alpha : A \longrightarrow R$ and $\beta : B \longrightarrow R$ is the sized set

$$A \times B \longrightarrow \mathcal{R}; (a, b) \mapsto \alpha(a) + \beta(b).$$

The projection maps, π_0 and π_1, are bounded by identity polynomials. Given $f : C \longrightarrow A$ and $g : C \longrightarrow B$, the tuple map $\langle f, g \rangle$ is bounded by $p + q$, where p is a bound for f and q is a bound for g. The terminal object is the $\mathcal{R}$-sized set $\tau : \{*\} \longrightarrow \mathcal{R}; * \mapsto 0$.

The category of $\mathcal{R}$-sized sets and maps with a constant bound, denoted $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$, is the subcategory of $\mathcal{R}$-$\mathsf{Set}_{\mathsf{poly}}$ consisting of $\mathcal{R}$-sized sets and functions $f : A \longrightarrow B$ such that there exists a *constant* $c \in \mathcal{R}$ satisfying: for all $a \in A$, $\beta(f(a)) \leq \alpha(a) + c$.

Proposition 2.6 $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$ *is an* $\mathcal{R}$-$\mathsf{Set}_{\mathsf{poly}}$*-strong category.*

Proof. It remains to describe the module structure. A cross map $(A, B) \longrightarrow C$ is a function $f : A \times B \longrightarrow C$ such that there exists a polynomial bound $p \in \mathcal{R}[x]$ satisfying, for all $(a, b) \in A \times B$, $\gamma(f(a, b)) \leq p(\alpha(a)) + \beta(b)$.

The strong composition is defined as follows: given $f : (A_1, B_1) \longrightarrow B_2$ and $g : (A_2, B_2) \longrightarrow C$ bounded by p and q, respectively, their composite is defined by $(f; g)((a_1, a_2), b_1) = g(a_2, f(a_1, b_1))$ and is bounded by $p + q$ as:

$$\begin{aligned}
\gamma((f; g)((a_1, a_2), b_1)) &\leq \gamma(g(a_2, f(a_1, b_1))) \\
&\leq q(\alpha_2(a_2)) + \beta_2(f(a_1, b_1)) \\
&\leq q(\alpha_2(a_2)) + p(\alpha_1(a_1)) + \beta_1(b_1) \\
&\leq (p + q)(\alpha(a_1, a_2)) + \beta_1(b_1)
\end{aligned}$$

The identity cross maps $i_A : \mathbf{1} \times A \longrightarrow A$ are given by second projection and are bounded by 0. □

3 Affine structure, products and coproducts

The $\mathbb{X}$-strong categories we are interested in here have much more structure: the player category is affine closed [10] with products and coproducts. This section describes how this structure is defined for $\mathbb{X}$-strong categories and gives the corresponding fibrational interpretation. Finally, the corresponding structure in $\mathcal{R}$-sized sets is described.

Recall that if we wish to capture Bellantoni and Cook's or Leivant's system of PTIME we would have to forgo the closure at this stage. As mentioned in the introduction we include closure in order to obtain a more expressive recursion principle.

---

[10] Affine closed in the sense that it is a symmetric monoidal closed category in which the tensor unit is terminal.

3.1 Interpretation in $\mathbb{X}$-strong categories

An $\mathbb{X}$-strong category $\mathbb{Y}$ is **affine closed** in case $\mathbb{Y} = (\mathbb{Y}, \otimes, \multimap, \mathbf{1})$ is affine closed and this structure extends to the module:

$$\frac{(X, Y_1) \xrightarrow{f} Y_1' \quad (X, Y_2) \xrightarrow{g} Y_2'}{(X, Y_1 \otimes Y_2) \xrightarrow{f \otimes g} Y_1' \otimes Y_2'} \qquad \frac{(X, Z \otimes Y_1) \xrightarrow{f} Y_2}{(X, Z) \xrightarrow{\mathsf{cur}(f)} Y_1 \multimap Y_2}$$

These must satisfy the equations:

- The tensor product is an $\mathbb{X}$-strong bifunctor: $(f \otimes g); (f' \otimes g') = f; f' \otimes g; g'$, $i_{Y_1} \otimes i_{Y_2} = i_{Y_1 \otimes Y_2}$. The monoidal natural isomorphism $a_\otimes, u_\otimes^L, u_\otimes^R, c_\otimes$ are $\mathbb{X}$-strong natural transformations: e.g. $(1, a_\otimes)((f \otimes g) \otimes h) = (f \otimes (g \otimes h))a_\otimes$, etc;

- The tensor product must behave well with the module structure: $(x, y_1 \otimes y_2)(f \otimes g) = (x, y_1)f \otimes (x, y_1)g$ and $(f \otimes g)(y_1 \otimes y_2) = fy_1 \otimes gy_2$;

- For the closed structure: $(\mathsf{cur}(f) \otimes (!_X, 1)i_A)\mathsf{ev} = f$. We also assume $(x, 1)\mathsf{cur}(f) = \mathsf{cur}((x, 1)f)$. And $h; \mathsf{cur}(f) = \mathsf{cur}((h \otimes (!, 1)i); f)$ and $\mathsf{cur}((1, \mathsf{ev})i_B) = i_{A \multimap B}$.

An $\mathbb{X}$-strong category $\mathbb{Y}$ is **cartesian** in case $\mathbb{Y} = (\mathbb{Y}, \times, \mathbf{1})$ is cartesian and the cartesian structure extends to cross maps as well:

$$\frac{(X, Y) \xrightarrow{f} Y_1 \quad (X, Y) \xrightarrow{g} Y_2}{(X, Y) \xrightarrow{\langle f, g \rangle} Y_1 \times Y_2} \qquad \frac{}{(X, Y) \xrightarrow{!_{X,Y}} \mathbf{1}}$$

These must satisfy the following equations:

- $\langle f, g \rangle \pi_0 = f$, $\langle f, g \rangle \pi_1 = g$, and $\langle h\pi_0, h\pi_1 \rangle = h$.

- The terminal object satisfies: for any cross map $f : (X, Y) \to \mathbf{1}$, $f = !_{X,Y}$.

An $\mathbb{X}$-strong category $\mathbb{Y}$ has **coproducts** in case the categories $\mathbb{X}$ and $\mathbb{Y}$ have coproducts which are distributive with respect to the product in $\mathbb{X}$ and with respect to the tensor in $\mathbb{Y}$ (this latter is forced if $\mathbb{Y}$ is closed affine). This means there is a (strength) map $d : Z \times (Y_1 + Y_2) \to Z \times Y_1 + Z \times Y_2$ which is inverse to the natural map in the reverse direction. In addition we require that the coproducts work across the module in two ways:

$$\frac{(X_1, Y) \xrightarrow{h_1} X' \quad (X_2, Y) \xrightarrow{h_2} Y'}{(X_1 + X_2, Y) \xrightarrow[\langle h_1 | h_2 \rangle]{} Y'} \qquad \frac{(X, Y_1) \xrightarrow{h_1} X' \quad (X, Y_2) \xrightarrow{h_2} Y'}{(X, Y_1 + Y_2) \xrightarrow[\langle h_1 | h_2 \rangle]{} Y'}$$

It is worth mentioning that we have not demanded that the products in $\mathbb{Y}$ distribute over the coproducts. This is quite deliberate as, although it is a

very natural requirement, letting higher-order types distribute over products allows the expression of PSPACE complete problems; this is explained further in [20] and is based on an observation of Hofmann [11]. This distributive law, however, is present in the example based on $\mathcal{R}$-sized sets – this should be expected as they are at least a PSPACE setting.

3.2   The fibrational interpretation

Products and coproducts can, of course, be defined at the 2-categorical level by demanding left and right adjoints to the diagonal $\mathbb{X}$-strong functor. The above presentation is a more explicit but equivalent formulation. The functorial presentation, however, has the advantage that it can be transported along the 2-functor which preserves products: and bun is of this form. This leads to (part of) the following equivalent fibrational statement:

Proposition 3.1 *Let $\mathbb{Y}$ be an affine closed $\mathbb{X}$-strong_category with products and coproducts. Then the corresponding fibration $p : \widetilde{\mathbb{Y}} \rightarrow \mathbb{X}$ is fibered affine closed and has fibered products and coproducts. This means that each of the fibers is an affine closed category possessing products and coproducts, and this structure is preserved on the nose by the reindexing functors.*

Proof. (Sketch:) In the fiber over $X \in \mathbb{X}$ the products are given by:

$$(X, Y_1) \times (X, Y_2) = (X, Y_1 \times Y_2)$$

The terminal object in $p^{-1}(X)$ is $(X, \mathbf{1})$ and the unique map to it from any object (X, Y) is of just $(1_X, !_Y)$.

Similarly, if an $\mathbb{X}$-strong category $\mathbb{Y}$ has coproducts, then so do the fibers in the corresponding fibration:

$$(X, Y_1) + (X, Y_2) = (X, Y_1 + Y_2)$$

with injection $(1_X, (1, \sigma_0) i_{Y_1 + Y_2})$ and $(1_X, (1, \sigma_1) i_{Y_1 + Y_2})$. Given maps $(1_X, h_1) : (X, Y_1) \rightarrow (X, Z)$ and $(1_X, h_2) : (X, Y_2) \rightarrow (X, Z)$ the cotuple map is defined by $(1_X, \langle h_1 | h_2 \rangle) : (X, Y_1 + Y_2) \rightarrow (X, Z)$.

If $\mathbb{Y}$ has distributive coproducts then then the isomorphism $d : Z \times (Y_1 + Y_2) \rightarrow (Z \times Y_1) + (Z \times Y_2)$ lifts to the fiber $p^{-1}(X)$ as $(1_X, (!, d) i)$.

The affine closed structure lifts similarly:

$$(X, Y_1) \otimes (X, Y_2) = (X, Y_1 \otimes Y_2)$$
$$(X, Y_1) \multimap (X, Y_2) = (X, Y_1 \multimap Y_2).$$

Given a map $(1_X, h) : (X, Y_1 \otimes Z) \rightarrow (X, Y_2)$ its curried form is the map $(1_X, \mathsf{cur}(h)) : (X, Z) \rightarrow (X, Y_1 \multimap Y_2)$.

Finally, we note that the cartesian maps are of the form $(X_1, Y) \longrightarrow (X_2, Y)$ with the second component fixed, thus, the reindexing functors preserve all of this structure strictly. $\qquad\qquad\square$

It follows that the total category $\widetilde{\mathbb{Y}}$ is itself cartesian (as the fibration is fibered cartesian and the base is cartesian). However, it does not, in general, inherit the coproduct structure.

3.3 The interpretation in $\mathcal{R}$-sized sets

In order to describe this additional structure in $\mathcal{R}$-sized sets it is necessary to assume that the size rig $\mathcal{R}$ has infima for all non-empty sets and these are preserved by the operations. Note that this means that the maximum of any two elements in the rig can be defined by $\max(a, b) := \inf\{c \mid a, b \leq c\}$ as $a + b \in \{c \mid a, b \leq c\}$. Note that both $\mathbb{N}$ and $\mathbb{R}_{\geq 0}$ are still examples. With this additional assumption on $\mathcal{R}$, we have:

Proposition 3.2 $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$ *is affine closed.*

Proof. The tensor structure on $\mathcal{R}$-sized sets is defined as follows. Given $\alpha : A \to \mathcal{R}$ and $\beta : B \to \mathcal{R}$ define

$$(\alpha \otimes \beta) : A \times B \longrightarrow \mathcal{R}; (a, b) \mapsto \alpha(a) + \beta(b).$$

The tensor product of maps $f : A_1 \to B_1$ and $g : A_2 \to B_2$ is defined by $(f \otimes g)(a_1, a_2) = (f(a_1), g(a_2))$ and is bounded by the sum of the constant bounds for f and g. The tensor is affine as the tensor unit is the same as the terminal object.

Given $\alpha : A \to \mathcal{R}$ and $\beta : B \to \mathcal{R}$, the internal hom is defined by $\alpha \multimap \beta : C(A, B) \longrightarrow \mathcal{R}$, where $C(A, B)$ is the set of constant $\mathcal{R}$-sized maps from A to B, and

$$(\alpha \multimap \beta)(f) = \inf\{c \mid \forall a \in A.\beta(f(a)) \leq \alpha(a) + c\}.$$

Given a map $f : A \times X \to B$ of $\mathcal{R}$-sized sets bounded by $c \in \mathcal{R}$, define $\mathsf{cur}(f) : X \to C(A, B)$ by $x \mapsto \lambda a.f(a, x)$. This is bounded by the same constant $c \in \mathcal{R}$ as:

$$
\begin{aligned}
(\alpha \multimap \beta)(\mathsf{cur}(f)(x)) &= \inf\{c' \mid \forall a \in A.\beta(\mathsf{cur}(f)(x)(a)) \leq \alpha(a) + c'\} \\
&= \inf\{c' \mid \forall a \in A.\beta(f(a, x)) \leq \alpha(a) + c'\} \\
&\leq \xi(x) + c
\end{aligned}
$$

since $\beta(f(a, x)) \leq \alpha(a) + \xi(x) + c$. The evaluation map $\mathsf{ev} : (A \multimap B) \times A \to B$ is given by $\mathsf{ev}(f, a) = f(a)$ and is bounded as there exists a constant

13

$c \in \mathcal{R}$ such that:

$$\beta(f(a)) \leq \alpha(a) + c$$
$$\leq \alpha(a) + (\alpha \multimap \beta)(f) + c$$

For each $\mathcal{R}$-sized set X define $C_X(A, B)$ to be the collection of functions $f : X \times A \longrightarrow B$ such that there exists $p \in \mathcal{R}[x]$ satisfying, for all $a \in A$ and $x \in X$, $\beta(f(x, a)) \leq p(\xi(x)) + \alpha(a)$. As before, arbitrary infima in $\mathcal{R}$ are required to assign a size to elements of $C_X(A, B)$ and to make this into an internal hom object. $\qquad \square$

We now turn to the product structure for this example:

Proposition 3.3 $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$ *is cartesian.*

Proof. Given $\mathcal{R}$-sized sets $\alpha : A \longrightarrow \mathcal{R}$ and $\beta : B \longrightarrow \mathcal{R}$, their cartesian product is given by the function

$$(\alpha \times \beta) : A \times B \longrightarrow \mathcal{R}; (a, b) \mapsto \max(\alpha(a), \beta(b)).$$

The tuple of maps $f : A \longrightarrow B$ and $g : A \longrightarrow C$, bounded by constants c_1 and c_2 respectively, is the map $\langle f, g \rangle : A \longrightarrow B \times C$, and is bounded by $\max(c_1, c_2)$. Projections are given by the usual projection functions and are bounded by 0.

This structure naturally extends to the cross maps as follows. Given $f : A \times B \longrightarrow C_1$ and $g : A \times B \longrightarrow C_2$, bounded by $p, q \in R[x]$, respectively, the pairing map $\langle f, g \rangle$ is bounded by $\max(p, q)$ as, for all a:

$$\gamma(\langle f, g \rangle(a, b)) \leq \max(\gamma_1(f(a, b)), \gamma_2(g(a, b)))$$
$$\leq \max(p(\alpha(a)) + \beta(b), q(\alpha(a)) + \beta(b))$$
$$\leq \max(p(\alpha(a)), q(\alpha(a))) + \beta(b)$$

All of the equations are satisfied because they are satisfied in the underlying set-theoretic interpretation. $\qquad \square$

Notice the diagonal map $d : A \longrightarrow A \otimes A$ is not in general bounded by a constant, so the tensor product and the cartesian product are not isomorphic in $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$. However, the two *are* isomorphic in the category $\mathcal{R}$-$\mathsf{Set}_{\mathsf{poly}}$, as the diagonal can be bounded by the polynomial $2x \in \mathcal{R}[x]$.

Proposition 3.4 *The polarized strong category* $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$ *has coproducts.*

Proof. Recall that this means that both the category $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$ and the category $\mathcal{R}$-$\mathsf{Set}_{\mathsf{poly}}$ have coproducts and that the coproduct acts on the module maps in two different ways. The coproduct of $\mathcal{R}$-sized sets $\alpha : A \longrightarrow \mathcal{R}$ and $\beta : B \longrightarrow \mathcal{R}$ is defined as $\langle \alpha | \beta \rangle : A + B \longrightarrow \mathcal{R}$, where $\langle \alpha | \beta \rangle(l, a) = \alpha(a)$ and $\langle \alpha | \beta \rangle(r, b) = \beta(b)$, where we have used '$l$' to denote the left component and

'r' to denote the right component of the disjoint union of A and B. The injections are the usual set-theoretic ones and are clearly bounded. Given $f : A \to C$ and $g : B \to C$, bounded by p and q, respectively, the copairing map $\langle f|g \rangle : A + B \to \mathcal{R}$ is bounded by $p + q$. Both categories have distributive coproducts as the map $d : A \times (B_1 + B_2) \to (A \times B_1) + (A \times B_2)$, defined by $d(a, (i, b)) = (i, (a, b))$, is trivially bounded and is an isomorphism. Given cross maps $A \times B_1 \to C$ and $A \times B_2 \to C$, bounded by $p + c_1$ and $q + c_2$, respectively, the cotuple map $\langle f|g \rangle : A \times (B_1 + B_2) \to C$ is defined by $\langle f|g \rangle(a, (l, b)) = f(a, b)$ and $\langle f|g \rangle(a, (r, b)) = g(a, b)$ and is again bounded by $p + q + \max(c_1, c_2)$. $\qquad\square$

4 Lifting and comprehension

An $\mathbb{X}$-strong category has a "lifting" if there is an $\mathbb{X}$-strong functor from the player category $\mathbb{Y}$ to the opponent category $\mathbb{X}$ which satisfies certain properties. Lifting plays an important role in the recursion principle described in the next section. It also has an appealing fibrational interpretation as it corresponds to the bundle fibration having "comprehension".

4.1 Lifting in $\mathbb{X}$-strong categories

We say that the module has a lift if for each $Y \in \mathbb{Y}$ there is an object $\uparrow(Y) \in \mathbb{X}$ and a module map

$$(\uparrow(Y), \mathbf{1}) \xrightarrow{\epsilon_Y} Y$$

such that whenever $(X, \mathbf{1}) \xrightarrow{h} Y$ is a module map there is a unique map $h^\flat : X \to \uparrow(Y)$ making

$$\begin{array}{ccc} (X, \mathbf{1}) & \xrightarrow{\ h\ } & Y \\ {\scriptstyle (h^\flat, 1)}\big\downarrow & {\scriptstyle \epsilon_Y} \nearrow & \\ (\uparrow(Y), \mathbf{1}) & & \end{array}$$

commute. The combinator $\flat$ is an operation which takes certain cross maps to $\mathbb{X}$-maps. We can define an operation in the other direction $\sharp$ by $g^\sharp = (g, 1)\epsilon_Y$. Then the following equations are easy consequences of the definition:

$$(x^\sharp)^\flat = x$$
$$(h^\flat)^\sharp = h$$

We also have $((x, 1)hy)^\flat = xh^\flat(\epsilon y)^\flat$.

Proposition 4.1 *Lifting defines an $\mathbb{X}$-strong functor $\uparrow(-) : \mathbb{Y} \to \mathbb{X}$ defined by $Y \mapsto \uparrow(Y)$ and $y : Y_1 \to Y_2 \mapsto (\epsilon_{Y_1} y)^\flat$.*

Proof. The identity is preserved as $\uparrow (1_Y) = (\epsilon_Y 1_Y)^\flat = (1_{\uparrow(Y)}^\sharp)^\flat = 1_{\uparrow(Y)}$. Composition is preserved as:

$$
\begin{aligned}
\uparrow(y_1)\uparrow(y_2) &= (\epsilon_{Y_1} y_1)^\flat (\epsilon_{Y_2} y_2)^\flat \\
&= (((\epsilon_{Y_1} y_1)^\flat (\epsilon_{Y_2} y_2)^\flat)^\sharp)^\flat \\
&= (((\epsilon_{Y_1} y_1)^\flat (\epsilon_{Y_2} y_2)^\flat, 1)\epsilon_{Y_3})^\flat \\
&= (((\epsilon_{Y_1} y_1)^\flat, 1)((\epsilon_{Y_2} y_2)^\flat, 1)\epsilon_{Y_3})^\flat \\
&= (((\epsilon_{Y_1} y_1)^\flat, 1)\epsilon_{Y_2} y_2)^\flat \\
&= (\epsilon_{Y_1} y_1 y_2)^\flat \\
&= \uparrow(y_1 y_2)
\end{aligned}
$$

This shows that we have a mere functor from $\mathbb{Y}$ to $\mathbb{X}$. To show it is $\mathbb{X}$-strong we must define it on cross maps: Given a cross map $h : (X, Y_1) \to Y_2$ we define $(\epsilon_{Y_1}; h)^\flat : (\uparrow (Y_1) \times X) \to \uparrow (Y_2)$. First we show that this behaves correctly with the module structure:

$$
\begin{aligned}
(\epsilon; (hy))^\flat &= ((\epsilon; h)y)^\flat \\
&= (\epsilon; h)^\flat (\epsilon y)^\flat
\end{aligned}
$$

and:

$$
\begin{aligned}
(\epsilon; (x, y)h)^\flat &= ((1 \times x, 1)(\epsilon y; h))^\flat \\
&= (1 \times x)(\epsilon y; h)^\flat \\
&= (1 \times x)(((\epsilon y)^\flat, 1)\epsilon; h)^\flat \\
&= (1 \times x)(((\epsilon y)^\flat \times 1, 1)(\epsilon; h))^\flat \\
&= (1 \times x)((\epsilon y)^\flat \times 1)(\epsilon; h)^\flat \\
&= ((\epsilon y)^\flat \times x)(\epsilon; h)^\flat
\end{aligned}
$$

Next we show that this preserves the identity and module composition. For the identity we have:

$$
\begin{aligned}
(\epsilon_Y; i_Y)^\flat &= ((\pi_0, 1)\epsilon_Y)^\flat \\
&= \pi_0 : Y \times \mathbf{1} \to Y
\end{aligned}
$$

which is the identity in $\mathbb{X}$ considered an $\mathbb{X}$-strong category. Next we check that it preserves the module composition:

$$
\begin{aligned}
(\epsilon_{Y_1}; f)^\flat; (\epsilon_{Y_2}; g)^\flat &= a_\times((\epsilon_{Y_1}; f)^\flat \times 1)(\epsilon_{Y_2}; g)^\flat \\
&= a_\times((((\epsilon_{Y_1}; f)^\flat \times 1), 1)\epsilon_{Y_2}; g)^\flat \\
&= a_\times((((\epsilon_{Y_1}; f)^\flat, 1)\epsilon_{Y_2}); g)^\flat \\
&= a_\times((\epsilon_{Y_1}; f); g)^\flat
\end{aligned}
$$

$$= a_\times((a_\times^{-1}, 1)\epsilon_{Y_1}; (f; g))^\flat$$
$$= (\epsilon_{Y_1}; (f; g))^\flat$$

$\square$

This allows one to define the **lift combinator**:

$$\frac{(X, Y \otimes Y') \xrightarrow{f} Y''}{(X \times \uparrow(Y), Y') \xrightarrow[f^\uparrow]{} Y''}$$

as follows:

$$\frac{\dfrac{(\uparrow(Y), \mathbf{1}) \xrightarrow{\epsilon_Y} Y \quad (\uparrow(Y), Y') \xrightarrow{(!, 1)i_{Y'}} Y'}{(\uparrow(Y), Y') \xrightarrow{\epsilon_{Y,Y'}} Y \otimes Y' \qquad (X, Y \otimes Y') \xrightarrow{f} Y''}}{\dfrac{(\uparrow(Y) \times X, Y') \xrightarrow{\epsilon_{Y,Y'}; f} Y''}{(X \times \uparrow(Y), Y') \xrightarrow[f^\uparrow]{} Y''}}$$

with $\epsilon_{Y,Y'} = (1, u_L^{-1})(e_Y \otimes (!, 1)i_{Y'})$ and $f^\uparrow = (c_\times, 1)(\epsilon_{Y,Y'}; f)$. Clearly, a cartesian category $\mathbb{X}$ regarded as a $\mathbb{X}$-strong category has a trivial lift given by the identity functor.

Lemma 4.2 *Lifting is iso-monoidal*[11] *for the products and tensor in $\mathbb{Y}$ onto the product is $\mathbb{X}$.*

Proof. Because the product and tensor are affine we expect comonoidal maps $\uparrow(Y_1 \otimes Y_2) \to \uparrow(Y_1) \times \uparrow(Y_2)$ and $\uparrow(Y_1 \times Y_2) \to \uparrow(Y_1) \times \uparrow(Y_2)$. The maps in the reverse direction are given by:

$$\frac{\dfrac{\dfrac{\dfrac{\overline{Y_1 \otimes Y_2 \to Y_1 \otimes Y_2}}{(\mathbf{1}, Y_1 \otimes Y_2) \to Y_1 \otimes Y_2}}{(\uparrow(Y_1), Y_2) \to Y_1 \otimes Y_2}}{(\uparrow(Y_1) \times \uparrow(Y_2), \mathbf{1}) \to Y_1 \otimes Y_2}}{\uparrow(Y_1) \times \uparrow(Y_2) \to \uparrow(Y_1 \otimes Y_2)} \qquad \frac{\dfrac{\dfrac{\dfrac{\overline{Y_1 \to Y_1}}{(\mathbf{1}, Y_1) \to Y_1}}{(\uparrow(Y_1), \mathbf{1}) \to Y_1}}{(\uparrow(Y_1) \times \uparrow(Y_2), \mathbf{1}) \to Y_1} \quad \dfrac{\dfrac{\dfrac{\overline{Y_2 \to Y_2}}{(\mathbf{1}, Y_2) \to Y_2}}{(\uparrow(Y_2), \mathbf{1}) \to Y_2}}{(\uparrow(Y_1) \times \uparrow(Y_2), \mathbf{1}) \to Y_2}}{\dfrac{(\uparrow(Y_1) \times \uparrow(Y_2), \mathbf{1}) \to Y_1 \times Y_2}{\uparrow(Y_1) \times \uparrow(Y_2) \to \uparrow(Y_1 \times Y_2)}}$$

Moreover, as $\mathbf{1}$ is terminal there is a unique map $!_{\uparrow(\mathbf{1})} : \uparrow(\mathbf{1}) \to \mathbf{1}$ which is inverse to the map $(i_{\mathbf{1}})^\flat : \mathbf{1} \to \uparrow(\mathbf{1})$. $\square$

It is also important for our purpose that the lift preserves the coproduct structure as well and we shall simply *demand* that this is the case. That is, we shall demand that the canonical monoidal map is in fact an isomorphism.

---

[11] These are often called *strong* monoidal, however, the reader will appreciate that we have quite a few strong notions in this paper already!

4.2 Lifting in fibrations: comprehension

Recall that a fibration $p : \mathbb{Y} \to \mathbb{X}$, which has the terminal object functor $T : \mathbb{X} \to \mathbb{Y}$, admits *comprehension* if this functor has a right adjoint [8].

Proposition 4.3 *If an $\mathbb{X}$-strong category $\mathbb{Y}$ has a lift, then the corresponding fibration $p : \widetilde{\mathbb{Y}} \to \mathbb{X}$ admits comprehension in the above sense.*

Proof. Let $\mathbb{Y}$ be an $\mathbb{X}$-strong category with a lift operator. Then there is a functor $R : \widetilde{\mathbb{Y}} \to \mathbb{X}$ defined by $R(X, Y) = X \times \uparrow(Y)$ and $R(x, h) = \langle \pi_0 x, (h; \epsilon)^\flat \rangle$. We claim that this is right adjoint to the terminal object functor $T : \mathbb{X} \to \widetilde{\mathbb{Y}}$, defined by $T(X) = (X, \mathbf{1})$ and $T(x) = (x, !)$. I.e. that there is a bijection:

$$\frac{(X, \mathbf{1}) \to (X', Y)}{X \to X' \times \uparrow(Y)}$$

The unit and counit of the adjunction are:

$$\eta_X = \langle 1_X, !_X \rangle : X \to X \times \uparrow(\mathbf{1})$$

$$\epsilon_{X,Y} = (\pi_0, (\pi_1, 1)\epsilon_Y) : (X \times \uparrow(Y), \mathbf{1}) \to (X, Y)$$

These are certainly natural transformations so it remains to verify the adjunction equations:

$$\begin{aligned}
T\eta; \epsilon T &= (\langle 1_X, !_X \rangle, !); (\pi_0, (\pi_1, 1)\epsilon_\mathbf{1}) \\
&= (1_X, (\Delta, 1)(!; (\langle 1_X, !_X \rangle, 1)(\pi_1, 1)\epsilon_\mathbf{1})) \\
&= (1_X, !)
\end{aligned}$$

and:

$$\begin{aligned}
\eta R R \epsilon &= \langle 1_{X \times \uparrow(Y)}, !_{X \times \uparrow(Y)} \rangle \langle \pi_0 \pi_0, ((\pi_1, 1)\epsilon_Y; \epsilon_\mathbf{1})^\flat \rangle \\
&= \langle 1_{X \times \uparrow(Y)}, !_{X \times \uparrow(Y)} \rangle \langle \pi_0 \pi_0, ((\pi_1, 1)\epsilon_Y; i_\mathbf{1})^\flat \rangle \\
&= \langle 1_{X \times \uparrow(Y)}, !_{X \times \uparrow(Y)} \rangle \langle \pi_0 \pi_0, ((\pi_0, 1)(\pi_1, 1)\epsilon_Y)^\flat \rangle \\
&= \langle 1_{X \times \uparrow(Y)}, !_{X \times \uparrow(Y)} \rangle \langle \pi_0 \pi_0, ((\pi_0 \pi_1, 1)\epsilon_Y)^\flat \rangle \\
&= \langle 1_{X \times \uparrow(Y)}, !_{X \times \uparrow(Y)} \rangle \langle \pi_0 \pi_0, \pi_0 \pi_1 \rangle \\
&= \langle 1_{X \times \uparrow(Y)}, !_{X \times \uparrow(Y)} \rangle \pi_0 \\
&= 1_{X \times \uparrow(Y)}
\end{aligned}$$

This completes the proof. □

4.3 Lifting in $\mathcal{R}$-sized sets

Lifting has a very simple interpretation in the category of $\mathcal{R}$-sized sets.

Proposition 4.4 *The $\mathcal{R}$-$\mathsf{Set}_{\mathsf{poly}}$-strong category $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$ has a lift.*

18

Proof. Lifting here is the identity on $\mathcal{R}$-sized sets and for any $\mathcal{R}$-sized set A there is a map $\epsilon_A : A \times \{*\} \longrightarrow A$ defined by $\epsilon_A(a, *) = a$. Then given any cross map $f : A \times \{*\} \longrightarrow B$, bounded by $p \in \mathcal{R}[x]$, there is a map $f^\flat : A \longrightarrow B$, uniquely defined by $f^\flat(a) = f(a, *)$, and is bounded by p as well. $\qquad\square$

5   Polarized operators and "comprehended" recursion

Data types in this setting correspond to "comprehended" initial fixed points for polarized operators. Polarized operators are more than $\mathbb{X}$-strong functors as they also act on cross maps:

$$\frac{(X, \mathbf{1}) \xrightarrow{h} Y'}{(F_o(X), \mathbf{1}) \xrightarrow[F_{op}(h)]{} F_p(Y')} \; F_{op}$$

A peculiarity of the (initial) fixed point calculus in his setting is that inductive data (i.e. fixed point data) does not in general supply material from which one can build more inductive data. This is because inductive data does not, in general, organize itself into a polarized operator.

   The recursion principle which we present here is the "circular" principle due to Varmo Vene [22] and Luigi Santocanale [21]. It is based on using a "circular" combinator to determine maps from inductive types. From the programming perspective this is quite natural as the style is similar to a definition by recursion.

   We show below that this circular recursion principle, when the player world is affine closed, is equivalent to a more usual looking initial algebra principle. Of course, one can also use the circular recursion principle when the player world is not affine closed: the result is a scheme which is strictly more expressive than the initial algebra scheme as it has some "built-in" higher-order.

   We illustrate the recursion principle by using it to count the leaves of a (Hofmann) tree and to build an exponential (Leivant) tree.

5.1   *Polarized operators*

A **polarized operator** F on an $\mathbb{X}$-strong category consists of a pair of strong functors $F_p : \mathbb{Y}^n \longrightarrow \mathbb{Y}$, and $F_o : \mathbb{X}^n \longrightarrow \mathbb{X}$, and a map of cross maps F_{op}:

$$\frac{(X_1, \mathbf{1}) \xrightarrow{f_1} Y_1 \quad \cdots \quad (X_n, \mathbf{1}) \xrightarrow{f_n} Y_n}{(F_o(X_1, \ldots, X_n), \mathbf{1}) \xrightarrow{F_{op}(f_1, \ldots, f_n)} F_p(Y_1, \ldots, Y_n)} \; F_{op}$$

satisfying various natural conditions. In the unary case these are:

- F_o and F_p are $\mathbb{X}$-strong functors;
- $F_{op}((x, 1)h) = (F_o(x), 1)F_{op}(h)$ and $F_{op}(hy) = F_{op}(h)F_p(y)$.

- When there is a lift, lifting must preserve the polarized in the sense that there is a strong natural isomorphism γ^F such that:

$$
\begin{array}{ccc}
\mathbb{Y} & \xrightarrow{\;\;\uparrow\;\;} & \mathbb{X} \\
{\scriptstyle F_p}\downarrow & \Downarrow \gamma^F & \downarrow{\scriptstyle F_o} \\
\mathbb{Y} & \xrightarrow[\;\;\uparrow\;\;]{} & \mathbb{X}
\end{array}
$$

In the initial settings it is usually the case that every object in the opponent world is actually a lifted player object. In such settings the polarized operators are completely determined by the player side. Thus, the specification of these operators in the player world is often the crucial aspect.

Below we list the polarized operators which are *always* present:

(i) For any object A in $\mathbb{Y}$, the constant functors $K_p^A : \mathbb{Y}^0 \to \mathbb{Y}$ and $K_o^A : \mathbb{X}^0 \to \mathbb{X}$, defined by $K_p^A(Y) = A$ and $K_o^A(X) = \uparrow (A)$, form a polarized operator. In this case $K_{op}^A(*) = (!_{\uparrow(A)}, 1)\epsilon_A$.

(ii) The tensor product in $\mathbb{Y}$ and the product in $\mathbb{X}$ form a polarized operator.

(iii) The product in $\mathbb{Y}$ and the product in $\mathbb{X}$ form a polarized operator.

(iv) The coproduct in $\mathbb{Y}$ and coproduct in $\mathbb{X}$ form a polarized operator (this is why we required lifting to preserve coproducts).

Polarized operators compose as operations on a polarized strong category. Thus, further examples can be generated from the above basic examples. Thus, any "polynomial polarized operator", generated by $+, \times, \otimes$ and constants, is a polarized operator.

5.2 The circular recursion principle

Let F be an n-ary polarized operator on a polarized strong category $\mathbb{Y}$. A **circular combinator**, $c[\_]$ is a pair of assignments:

$$
\frac{h : (X, Z \otimes D) \to B}{c_p[h] : (X, F_p(Z) \otimes D) \to B}
$$

$$
\frac{h : (X \times V, D) \to B}{c_o[h] : (X \times F_o(V), D) \to B}
$$

in which X, B and D are fixed and called respectively the opponent context, the player context, and the base. This data is a *combinator* in the sense that it satisfies:

$$
c_o[(\langle 1, v \rangle, (!, 1)i_D)h] = (\langle 1, F_o(v)\rangle, (!, 1)i_D)c_o[h]
$$
$$
c_p[(1, f \otimes (!, 1)i_D)h] = (1, F_p(f) \otimes (!, 1)i_D)c_p[h]
$$
$$
c_o[(\pi_0, r \otimes (!, 1)i_D)h] = (\pi_0, F_{op}(r) \otimes (!, 1)i_D)c_p[h]
$$

Proposition 5.1 *In an affine closed polarized strong category circular combinators with contexts X and D and base B are in bijective correspondence to maps*

$$(X, F_p(D \multimap B)) \xrightarrow[g]{} D \multimap B.$$

Proof. Given such a map g define the circular combinator by:

$$\frac{\dfrac{(X, Z \otimes D) \xrightarrow{f} B}{(X, Z) \xrightarrow{\mathsf{cur}(f)} D \multimap B}}{\dfrac{(X, F_p(Z)) \xrightarrow{F_p(\mathsf{cur}(f))} F_p(D \multimap B) \quad (X, F_p(D \multimap B)) \xrightarrow{g} D \multimap B}{\dfrac{(X, F_p(Z)) \xrightarrow{F_p(\mathsf{cur}(f))g} D \multimap B}{(X, F_p(Z) \otimes D) \xrightarrow[c_p[f] = ((F_p(\mathsf{cur}(f))g) \otimes i_D)\mathsf{eval}]{} B}}}$$

and

$$\frac{\dfrac{(X \times Z, D) \xrightarrow{f'} B}{(X \times Z, \mathbf{1}) \xrightarrow{\mathsf{cur}(f')} D \multimap B}}{\dfrac{(X \times F_o(Z), \mathbf{1}) \xrightarrow{\theta_X^F F_{op}(\mathsf{cur}(f'))} F_p(D \multimap B) \quad (X, F_p(D \multimap B)) \xrightarrow{g} D \multimap B}{\dfrac{(X \times F_o(Z), \mathbf{1}) \xrightarrow{\theta_X^F F_{op}(\mathsf{cur}(f'))g} D \multimap B}{(X \times F_o(Z), D) \xrightarrow[c_o[f'] = ((\theta_X^F F_{op}(\mathsf{cur}(f'))g) \otimes i_D)\mathsf{eval}]{} B}}}$$

Conversely take:

$$\frac{\dfrac{(X, (D \multimap B) \otimes D) \xrightarrow{\mathsf{ev}'} B}{(X, F_p(D \multimap B) \otimes D) \xrightarrow{c_p[\mathsf{ev}']} B}}{(X, F_p(D \multimap B)) \xrightarrow[\mathsf{cur}(c_p[\mathsf{ev}'])]{} D \multimap B}$$

$\square$

A **polarized inductive data type** in an affine polarized strong category for a polarized operator F is a fixed point $\mu y.F_p(y)$ of F_p, that is, there is an isomorphism:

$$\mathsf{cons} : F_p(\mu y.F_p(y)) \longrightarrow \mu y.F_p(y)$$

in $\mathbb{Y}$ together with the following (circular) universal property: given any circular combinator $c[\_]$ for F with contexts X and D over B then there is a unique map $\mu a.c[a] : (X \times \uparrow (\mu y.F_p(y)), D) \to B$ making the following trian-

gle commute:

$$(X \times F_o(\uparrow(\mu y.F_p(y)))), D) \xrightarrow{(1\times\uparrow(\mathsf{cons}),1)} (X\times \uparrow(\mu y.F_p(y)), D)$$

with lower maps $c_o[\mu a.c[a]]$ and $\mu a.c[a]$ to B.

Notice that in the affine *closed* case we can resolve this to get a more usual looking fixed point property using the above proposition:

$$(X \times F_o(\uparrow(\mu y.F_p(y))), \mathbf{1}) \xrightarrow{(1\times\uparrow(\mathsf{cons}),1)} (X\times \uparrow(\mu y.F_p(y)), \mathbf{1})$$

$$(1,F_{op}(\mathsf{cur}(\mu a.c[a])])) \downarrow \qquad\qquad \downarrow \mathsf{cur}(\mu a.c[a])$$

$$(X, F_p(D \multimap B)) \xrightarrow{\mathsf{cur}(c_p[(!,1)i\ \mathsf{ev}'])} D \multimap B$$

5.3 Counting the leaves of a tree

To illustrate how this works consider the following example. Let $T'_p(A,B,C) = A + B \otimes C$ (and so $T'_o(X,Y,Z) = X + Y \times Z$) then define $\mathsf{Tree}_p(A) = \mu y.A + y \otimes y$. To remove unnecessary clutter we shall replace the type variable A by a constant. The fixed point isomorphism can be resolved into two constructors

$$\mathsf{Leaf} : A_p \longrightarrow \mathsf{Tree}_p \qquad \mathsf{node} : \mathsf{Tree}_p \otimes \mathsf{Tree}_p \longrightarrow \mathsf{Tree}_p.$$

We shall also need the (unary) natural numbers $\mathsf{Nat} = \mu y.\mathbf{1} + y$ with the constructors $\mathsf{zero} : \mathbf{1} \longrightarrow \mathsf{Nat}$ and $\mathsf{Succ} : \mathsf{Nat} \longrightarrow \mathsf{Nat}$ in order to count.

The circular combinator will have empty o-context and both the p-context and the base Nat. This means we must define a combinator:

$$\frac{(\mathbf{1}, X \otimes \mathsf{Nat}) \xrightarrow{f} \mathsf{Nat}}{(\mathbf{1}, (A + X \otimes X) \otimes \mathsf{Nat}) \xrightarrow[c[f]]{} \mathsf{Nat}}$$

we define $c[f]$ as:

$$(\mathbf{1}, (A+X\otimes X)\otimes\mathsf{Nat}) \xrightarrow{(1,i\ \mathsf{d}_\otimes)} (\mathbf{1}, A\otimes\mathsf{Nat}+X\otimes X\otimes\mathsf{Nat}) \xrightarrow{\langle i\ !\mathsf{Succ}|(1\otimes f)f\rangle} \mathsf{Nat}$$

and then the map

$$(\uparrow(\mathsf{Tree}), \mathbf{1}) \xrightarrow{(1,(!,1)i\ \mathsf{Zero})} (\uparrow(\mathsf{Tree}), \mathsf{Nat}) \xrightarrow{\mu x.c[x]} \mathsf{Nat}$$

counts the leaves of the tree.

One can also build the exponential size Leivant style tree, however, we must use a slightly different datatype (and this change is enough to remove

the ability to count the leaves): $\mathsf{LTree} = \mu y.A + y \times y$ which has constructors LLeaf and LNode. Note that nodes are now *products* of trees which one should regard as being lazy. For this tree we can construct the combinator:

$$\frac{(\mathbf{1}, X) \xrightarrow{h} \mathsf{LTree}}{(\mathbf{1}, 1 + X) \xrightarrow[d[h]]{} \mathsf{LTree}}$$

by defining $d[h]$ as $(\mathbf{1}, 1 + X) \xrightarrow{\langle i \; \mathsf{LLeaf} | f \; \Delta \; \mathsf{LNode} \rangle} \mathsf{LTree}$. This builds an exponential size tree (but based on products).

5.4   Comprehended recursion in $\mathcal{R}$-sized sets

$\mathcal{R}$-sized sets have comprehended recursion:

Proposition 5.2 *The $\mathcal{R}$-$\mathsf{Set}_{\mathsf{poly}}$-strong category $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$ has all "polynomial" polarized inductive data types.*

Proof. Let $F(Z) = F_1(Z) + \cdots + F_k(Z)$ be a polarized operator generated by constants, coproducts, products and tensors in disjunctive normal form – which makes sense, in this setting, as both products and tensors distribute over coproducts. Such a functor always has a fixed point in Sets, F^*, which is the free algebra generated by the constructors cons_i. The size $\phi : F^* \to \mathcal{R}$ is defined inductively by:

$$\begin{aligned}
\phi(\mathsf{cons}_i(d)) &= 1 + \phi(d) \\
\phi(d_1; \ldots; d_n) &= \max(\phi(d_1), \ldots, \phi(d_n)) \\
\phi(d_1, \ldots, d_n) &= \sum_{l=1}^{n} \phi(d_l) \\
\phi(a) &= \alpha(a) \quad \text{(where } \alpha : A \to \mathcal{R} \text{ is parametric)}
\end{aligned}$$

Each constructor increases the size of its input by 1, so they are certainly maps in the player category. The map cons above is the cotuple of these constructors, and so it too is bounded by a size increase of 1. The inverse map is non-size increasing and so is also in $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$. This shows that this object is a fixed point in $\mathcal{R}$-$\mathsf{Set}_{\mathsf{const}}$.

It is convenient, in order to bound the recursion principle to use the transformation of proposition 5.1 and derive the size bounds from the fixed point form of the map. This then has to be evaluated to obtain the map we actually want. It suffices to show we can polynomially bound the map fold recursively

defined by:

$$
\begin{array}{ccc}
(X \times F_o(F^*), \mathbf{1}) & \xrightarrow{(1,(!,1)i \text{ cons}}} & (X \times F^*, \mathbf{1}) \\
\langle \pi_0, \theta_X^F \rangle, !) \downarrow & & \downarrow \text{fold} \\
(X \times F_o(X \times F^*), \mathbf{1}) & & \\
(1, F_{op}(\text{fold})) \downarrow & & \\
(X, F_p(D \multimap B)) & \xrightarrow{g} & D \multimap B
\end{array}
$$

Clearly this depends on g which, being in the p-world, it has a size constant increase dependent only on the o-context, $P_g(\xi(x))$. Next consider the maps down the lefthand side: the first map is the strength and, depending on the form of F, will duplicate the context a number of times, as its effect is exactly the same as for F_p this is also a constant size increase bounded by a polynomial in the o-context, $P_\theta(\xi(x))$. As F is a polynomial functor the size of $F_{op}(\text{fold})$ is bounded by a constant added to the size of its parameters: the only subtly being that the one parameter shown may actually occur many times and thus the bound can be written in the form:

$$
\beta(F_{op}((x, \text{fold}(x, z)))) \leq k + \sum_{j=1,..,r} \beta(\text{fold}(x, z_i))
$$

where the z_j are the next largest recursive occurrences of the data type within z.

Thus there is a constant bound (in the context size) by which each constructor can increase the size. Thus, as $\phi(x)$ always exceeds the number of constructors, we have:

$$
\beta(\text{fold}(x, z)) \leq \phi(z) \cdot (k + P_\theta(\xi(x)) + P_g(\xi(x)))
$$

Giving a polynomial bound as required. $\qquad\square$

6 Conclusion

$\mathcal{R}$-sized sets are a delightfully concrete model of this setting which allow one to illustrate some of the peculiar properties expected of systems with lower complexity – that is below primitive recursive. However, $\mathcal{R}$-sized sets with polynomial bounds certainly include both PTIME and PSPACE functions and the paper begs the question of whether this system can be used to capture precisely PTIME or PSPACE.

These matters are addressed in [20] which is still work in progress. That document describes a full programming language and type system in support of the development of a PTIME programming language called **Pola** [3]. It is a more complex system than that described here. It uses a bunched logic

for the programs in the player world, has type inference, and supports both inductive and coinductive data. Coinductive data has destruction which is constant time, and thus, like the closed structure discussed in this document, is completely in the player world. The inductive data is essentially as discussed here (although a slightly more powerful recursion principle is used).

One can easily program PTIME Turing machines in this languiage: thus, it is certainly PTIME complete. If the distributive law for products over co-products is assumed for *all* data, then QSAT can be programmed (following the observation of Hofmann). Thus this language is also PSPACE complete. The arguments for soundness are by structural induction: they show first that the language (with the distributive law) is PSPACE sound and second that, if one drops the distributive law (for higher-order, coinductive, and universal types), that the language is PTIME sound (PTIME completeness is not affected).

An interesting aspect of the settings discussed here is that they do provide a native notion of equality of maps because the (inductive) data types come packaged with the universal property discussed above. Thus, this means the initial settings come with a term logic with a native notion of equality and it would be interesting to know how this relates to the various logics [12] which have been considered already for expressing PTIME and PSPACE.

References

[1] J-M. Andreoli. Focussing and proof construction. *Ann. Pure and Applied Logic*, 107(1) (2001) 131–163.

[2] S. Bellantoni & S. Cook. A new recursion-theoretic characterization of the polytime functions. *Computational Complexity*, 2:97–110, 1992.

[3] M.J. Burrell. Pola project page. http://projects.wizardlike.ca/projects/pola.

[4] R. Cockett & R. Seely. Polarized category theory, modules and game semantics. *Theory and Application of Categories*, 18 (2007) 4–101.

[5] R. Cockett & D. Spencer. Strong Categorical Datatypes I. *International Meeting on Category Theory 1991*, AMS, Canadian Mathematical Society Proceedings, 1992.

[6] J-Y Girard. A new constructive logic: classical logic. *Mathematical Structures in Computer Science*, 1(3) (1991) 255–296.

[7] M. Hamano & P. Scott. Annals of Pure and Applied Logic, 145 (2007) 276-313.

[8] C. Hermida & B. Jacobs. Structural induction and coinduction in a fibrational setting. *Information and Computation*, 145(2) (1998) 107–152.

[9] M. Hofmann & U. Dal Lago. Quantitative Models and Implicit Complexity. FSTTCS 2005: Proceedings of Foundations of Software Technology and Theoretical Computer Science, Hyderabad, India Springer Lecture Notes in Computer Science 3821 (2005) 189–200.

[10] M. Hofmann. Type systems for polynomial-time computation. Habilitation thesis, University of Darmstadt, 1999.

[11] M. Hofmann. Linear types and non-size-increasing polynomial time computation. *Information and Computation*, 183(1) (2003) 57-85.

25

[12] Neil Immerman, Guest Column: *Progress in Descriptive Complexity*, SIGACT NEWS 36(4) (2005) 24-35.

[13] A. Kock. Strong Functors and Monoidal Monads. *Archiv der Math.* 23: 113120, 1972.

[14] Olivier Laurent. Étude de la polarisation en logique, Université Aix-Marseille II, Thèse de Doctorat (2002).

[15] Olivier Laurent. Polarized Games. *Annals of Pure and Applied Logic* no 1–3, Vol. 130 (2004) 79–123.

[16] D. Leivant. Stratified Functional Programs and Computational Complexity. In *Proc. 20th IEEE Symp. on Principles of Programming Languages*, (1993) 325-333.

[17] D. Leivant & J.-Y. Marion. Ramified Recurrence and Computational Complexity II: Substitution and Poly-space. In *Proc. CSL'94, Springer LNCS*, 933 (1994) 486-500.

[18] J. Otto. *Complexity Doctrines*. Ph.D. thesis, McGill University, 1995.

[19] R. Paré & L. Román. Monoidal categories with natural numbers object. *Studia Logica*, 48(3) (1989) 361 − 376.

[20] B. Redmond. Notes on the logic of safety. In progress. Available at: http://projects.wizardlike.ca/documents/5

[21] L. Santocanale. A calculus of circular proofs and its categorical semantics. *Proc. FOSSACS 2002, Springer LNCS*, No. 2303, 2002.

[22] V. Vene. *Categorical programming with inductive and coinductive types*. PhD thesis, University of Tartu, 2000.

[23] Richard J. Wood, *Indicial methods for relative categories*. PhD Thesis, Dalhousie University, 1976.