POLYNOMIAL LENSES

BARTOSZ MILEWSKI

1. Motivation

Lenses seem to pop up in most unexpected places. Recently a new type of lens showed up as a set of morphisms between polynomial functors. This lens seemed to not fit the usual classification of optics, so it was not immediately clear that it had an existential representation using coends and, consequently a profunctor representation using ends. A profunctor representation of optics is of special interest since it lets us compose optics using standard function composition. In this post I will show how the polynomial lens fits into the framework of general optics.

2. Polynomial Functors

A polynomial functor in **Set** can be written as a sum (coproduct) of representables:

$$P(y) = \sum_{n \in N} s_n \times \mathbf{Set}(t_n, y)$$

The two families of sets, s_n and t_n are indexed by elements of the set N (in particular, you may think of it as a set of natural numbers, but any set will do). In other words, they are fibrations of some sets S and T over N. In programming we call such families *dependent types*. We can also think of these fibrations as functors from a discrete category $\mathcal{N}$ to **Set**.

Since, in **Set**, the internal hom is isomorphic to the external hom, a polynomial functor is sometimes written in the exponential form, which makes it look more like an actual polynomial or a power series:

$$P(y) = \sum_{n \in N} s_n \times y^{t_n}$$

or, by representing all sets s_n as sums of singlentons:

$$P(y) = \sum_{n \in N} y^{t_n}$$

I will also use the notation $[t_n, y]$ for the internal hom:

$$P(y) = \sum_{n \in N} s_n \times [t_n, y]$$

Polynomial functors form a category **Poly** in which morphisms are natural transformations.

Consider two polynomial functors P and Q. A natural transformation between them can be written as an end. Let's first expand the source functor:

$$\mathbf{Poly}\left(\sum_k s_k \times [t_k, -], Q\right) = \int_{y\colon \mathbf{Set}} \mathbf{Set}\left(\sum_k s_k \times [t_k, y], Q(y)\right)$$

The mapping out of a sum is isomorphic to a product of mappings:

$$\cong \prod_k \int_y \mathbf{Set}\left(s_k \times [t_k, y], Q(y)\right)$$

We can see that a natural transformation between polynomials can be reduced to a product of natural transformations out of monomials. So let's consider a mapping out of a monomial:

$$\int_y \mathbf{Set}\left(s \times [t, y], \sum_n a_n \times [b_n, y]\right)$$

We can use the currying adjunction:

$$\int_y \mathbf{Set}\left([t, y], \left[s, \sum_n a_n \times [b_n, y]\right]\right)$$

or, in **Set**:

$$\int_y \mathbf{Set}\left(\mathbf{Set}(t, y), \mathbf{Set}\left(s, \sum_n a_n \times [b_n, y]\right)\right)$$

We can now use the Yoneda lemma to eliminate the end. This will simply replace y with t in the target of the natural transformation:

$$\mathbf{Set}\left(s, \sum_n a_n \times [b_n, t]\right)$$

The set of natural transformation between two arbitrary polynomials $\sum_k s_k \times [t_k, y]$ and $\sum_n a_n \times [b_n, y]$ is called a polynomial lens. Combining the previous results, we see that it can be written as:

$$\mathbf{PolyLens}\langle s, t\rangle\langle a, b\rangle = \prod_{k \in K} \mathbf{Set}\left(s_k, \sum_{n \in N} a_n \times [b_n, t_k]\right)$$

Notice that, in general, the sets K and N are different.

Using dependent-type language, we can describe the polynomial lens as acting on a whole family of types at once. For a given value of type s_k it determines the index n. The interesting part is that this index and, consequently, the type of the focus a_n and the type on the new focus b_n depends not only on the type but also on the *value* of the argument s_k.

Here's a simple example: consider a family of node-counted trees. In this case s_k is a type of a tree with k nodes. For a given node count we can still have trees with a different number of leaves. We can define a poly-lens for such trees that focuses on the leaves. For a given tree it produces a counted

vector a_n of leaves and a function that takes a counted vector b_n (same size, but different type of leaf) and returns a new tree t_k.

3. Lenses and Kan Extensions

After publishing an Idris implementation of the polynomial lens, Iceland Jack made an interesting observation on Twitter: The sum type in the definition of the lens looks like a left Kan extension. Indeed, if we treat a and b as co-presheaves, the left Kan extension of a along b is given by the coend:

$$Lan_b a \cong \int^{n \colon \mathcal{N}} a \times [b, -]$$

A coend over a discrete category is a sum (coproduct), since the co-wedge condition is trivially satisfied.

Similarly, an end over a discrete category $\mathcal{K}$ becomes a product. An end of hom-sets becomes a natural transformation. A polynomial lens can therefore be rewritten as:

$$\prod_{k \in K} \mathbf{Set}\left(s_k, \sum_{n \in N} a_n \times [b_n, t_k]\right) \cong [\mathcal{K}, \mathbf{Set}](s, (Lan_b a) \circ t)$$

Finally, since the left Kan extension is the left adjoint of functor pre-composition, we get this very compact formula:

$$\mathbf{PolyLens}\langle s, t\rangle\langle a, b\rangle \cong [\mathbf{Set}, \mathbf{Set}](Lan_t s, Lan_b a)$$

which works for arbitrary categories $\mathcal{N}$ and $\mathcal{K}$ for which the relevant Kan extensions exist.

4. Existential Representation

A lens is just a special case of optics. Optics have a very general representation as existential types or, categorically speaking, as coends.

The general idea is that optics describe various modes of decomposing a type into the focus (or multiple foci) and the residue. This residue is an existential type. Its only property is that it can be combined with a new focus (or foci) to produce a new composite.

The question is, what's the residue in the case of a polynomial lens? The intuition from the counted-tree example tells us that such residue should be parameterized by both, the number of nodes, and the number of leaves. It should encode the shape of the tree, with placeholders replacing the leaves.

In general, the residue will be a doubly-indexed family c_{mn} and the existential form of poly-lens will be implemented as a coend over all possible residues:

$$\mathbf{Pl}\langle s, t\rangle\langle a, b\rangle \cong \int^{c_{ki}} \prod_{k \in K} \mathbf{Set}\left(s_k, \sum_{n \in N} a_n \times c_{nk}\right) \times \prod_{i \in K} \mathbf{Set}\left(\sum_{m \in N} b_m \times c_{mi}, t_i\right)$$

To see that this representation is equivalent to the previous one let's first rewrite a mapping out of a sum as a product of mappings:

$$\prod_{i \in K} \mathbf{Set}\left(\sum_{m \in N} b_m \times c_{mi}, t_i\right) \cong \prod_{i \in K} \prod_{m \in N} \mathbf{Set}\left(b_m \times c_{mi}, t_i\right)$$

and use the currying adjunction to get:

$$\prod_{i \in K} \prod_{m \in N} \mathbf{Set}\left(c_{mi}, [b_m, t_i]\right)$$

The main observation is that, if we treat the sets N and K as a discrete categories $\mathcal{N}$ and $\mathcal{K}$, a product of mappings can be considered a natural transformation between functors. Functors from a discrete category are just mappings of objects, and naturality conditions are trivial.

A double product can be considered a natural transformation from a product category. And since a discrete category is its own opposite, we can (anticipating the general profunctor case) rewrite our mappings as natural transformations:

$$\prod_{i \in K} \prod_{m \in N} \mathbf{Set}\left(c_{mi}, [b_m, t_i]\right) \cong [\mathcal{N}^{op} \times \mathcal{K}, \mathbf{Set}]\left(c_{=-}, [b_=, t_-]\right)$$

The indexes were replaced by placeholders. This notation underscores the interpretation of b as a functor (co-presheaf) from $\mathcal{N}$ to **Set**, t as a functor from $\mathcal{K}$ to **Set**, and c as a profunctor on $\mathcal{N}^{op} \times \mathcal{K}$.

We can therefore use the co-Yoneda lemma to eliminate the coend over c_{ki}. The result is that $\mathbf{Pl}\langle s, t\rangle\langle a, b\rangle$ can be wrtitten as:

$$\int^{c_{ki}} \prod_{k \in K} \mathbf{Set}\left(s_k, \sum_{n \in N} a_n \times c_{nk}\right) \times [\mathcal{N}^{op} \times \mathcal{K}, \mathbf{Set}]\left(c_{=-}, [b_=, t_-]\right)$$

$$\cong \prod_{k \in K} \mathbf{Set}\left(s_k, \sum_{n \in N} a_n \times [b_n, t_k]\right)$$

which is exactly the original polynomial-to-polynomial transformation.

5. Acknowledgments

I'm grateful to David Spivak, Jules Hedges and his collaborators for sharing their insights and unpublished notes with me, especially for convincing me that, in general, the two sets N and K should be different.