Three Views on **Org**

Sophie Libkind

March 21, 2024

I first learned about **Org** from David Spivak back in 2021, and it remains one of my favorite constructions in **Poly**. I love **Org** because it articulates one of the most fundamental features of living systems: that in a composite system, not only do the parts change over time but the interaction pattern between the parts changes as well.

In this note, we will show three different views on **Org**. The first is the original way that I learned it from David and lives directly in **Poly**. The second two are perspectives introduced to me by Toby Smithe and Matteo Capucci. Each of these perspectives shows how **Org** is a particular case of a more general construction.

1 Vanilla Org

Generally, I think about **Org** as an operad, but here we will introduce **Org** as a symmetric monoidal category. Its objects are the objects of **Poly** and a morphism $\mathbf{Org}(p, q)$ is a $[p, q]$-coalgebra, in other words a set of states S and a polynomial map $Sy^S \to [p, q]$. Its monoidal product is given by $\otimes$.

Example 1. *Suppose* $p_1, \cdots, p_n : \mathbf{Poly}$ *represent interfaces for my subordinates. The positions of* p_i *are the outputs of my* i*th subordinate. The directions of* p_i *are the inputs that I send my* i*th subordinate. Suppose that* q *represents the interface for my manager. The positions of* q *are what I output to my manager and the directions of* q *are the instructions I receive from my manager. Then a morphism from* $p_1 \otimes \cdots p_n$ *in* **Org** *is a* $[p_1 \otimes \cdots \otimes p_n, q]$*-coalgebra. Unraveling defintions, this morphism consists a set of states* S *(my possible states) and a three maps:*

- ***Read out.*** *Given my state and outputs from my subordinates, an output to my manager.*
- ***Read in.****Given my states, outputs from my subordinates, and instructions to my manager, inputs to my subordinates.*
- ***Update.*** *Given my states, outputs from my subordinates, and instructions to my manager, a new state for myself.*

Hence my state influences both how the subordinates talk to each other and how their outputs affect what I output to my manager. And critically it evolves!

Org gets its name from *organization* because its morphisms represent evolving organizations, in this example organizations of workers.

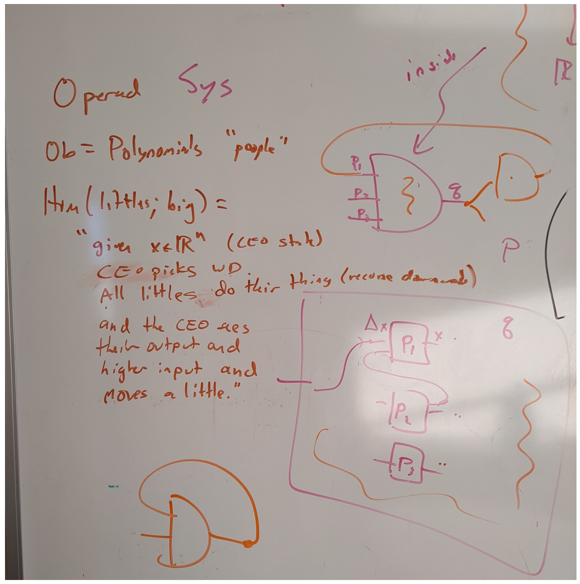


Figure 1: The whiteboard on which I first learned about **Org** (here named **Sys**).

2 Animating categories

Let's start with the abstraction of **animating categories** defined by Toby Smithe.

Let H be a category enriched in the symmetric monoidal category $(\mathcal{C}, \otimes_{\mathcal{C}}, 1_{\mathcal{C}})$ and let $\mathbf{Sys} : (\mathcal{C}, \otimes_{\mathcal{C}}, 1_{\mathcal{C}}) \to (\mathbf{Cat}, \times, 1)$ be a lax monoidal functor. Then we can pushforward H along $\mathbf{Sys}$ to get the category $\mathbf{Sys}_* H$ that is enriched in $(\mathbf{Cat}, \times, 1)$. [1] Therefore $\mathbf{Sys}_* H$ is a 2-category which Toby calls, **the category H animated by Sys**.

How is **Org** an animated category? First, note that since **Poly** has a $\otimes$ closure, there is a category $\mathbf{Poly}^{\mathcal{E}}$ that is enriched in $(\mathbf{Poly}, \otimes, y)$. In particular,

$$\mathsf{ob}\,\mathbf{Poly}^{\mathcal{E}} := \mathsf{ob}\,\mathbf{Poly}$$

and

$$\mathbf{Poly}^{\mathcal{E}}(p, q) := [p, q].$$

There is a lax monoidal functor $\mathbf{Coalg} : (\mathbf{Poly}, \otimes, y) \to (\mathbf{Cat}, \times, 1)$ which maps a polynomial p to the category of p-coalgebras. Unraveling the definitions, $\mathbf{Coalg}_* \mathbf{Poly}^{\mathcal{E}}$ is the 2-category whose objects are polynomials and where the morphisms from p to q are $[p, q]$-coalgebras. Sound familiar? [2]

[1] It's unclear whether this is enriched or weakly enriched and hence whether $\mathbf{Sys}_* H$ is a 2-category or a bicategory.

[2] Remember that **Org** is a symmetric monoidal category. What happened to its symmetric monoidal structure? Well instead of starting with a $\mathbf{Poly}^{\mathcal{E}}$ as a category enriched in **Poly**, we'll need to start with $\mathbf{Poly}^{\mathcal{E}}$ as a symmetric moniodal category enriched in **Poly**. Fortunately, I believe that the machinery developed by Brandon Shapiro gives us the tools to make sense of this statement.

3 Monads in Prof

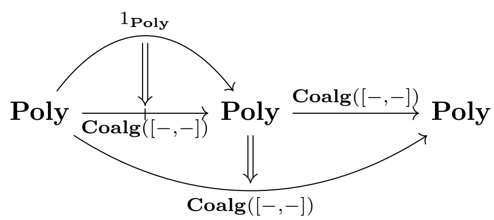
Recall the functor **Coalg** : **Poly** → **Cat** which sends each polynomial to the category of p-coalgebras. This is equivalent to a profunctor

$$1 \xrightarrow{\mathbf{Coalg}} \mathbf{Poly}.$$

But in fact we can generalize this to a profuctor[3]

$$\mathbf{Poly} \xrightarrow{\mathbf{Coalg}([-,-])} \mathbf{Poly}.^{4}$$

And the fun doesn't stop there! In fact $\mathbf{Coalg}([-,-])$ is a monad in **Prof** since we have maps as below that obey the monad laws.



Note that there is a functor from **Prof** → $\mathbf{Span}(\mathbf{Set})$ which sends a category to its set of objects. So $\mathbf{Coalg}([-,-])$ is a monad in $\mathbf{Span}(\mathbf{Set})$, in other words it's a category. What category? Why, **Org** of course!

[3]A detail to sort out: $\mathbf{Coalg}([-,-])$ in fact produces a *category* for each pair $p, q : Poly$. Therefore, we may in fact want the double category of polynomials as its domain and codomain as well as the category of double categories, double profunctors, and natural transforms.

[4]This is a generalization because **Coalg** is equivalent to $\mathbf{Coalg}([y,-])$.