1 Introduction

$\mathsf{Semantics} \dashv \mathsf{Syntax}$ relationships between some notion of type theory[1] and some flavor of category abound. They range from the well known and well studied example of the internal language of a topos, to linear type theory and $\star$-autonomous categories, to regular type theory for regular categories, among many others[2] and more recent developments, e.g. homtopy type theory, cubical type theory, etc. have extended this library of $\mathsf{Semantics} \dashv \mathsf{Syntax}$ relationships into higher category theory.

But what makes such a relationship, and therefor a type theory *practical*? Can we conjecture some criteria which judge a particular $\mathsf{Semantics} \dashv \mathsf{Syntax}$ relationship well adapted, both in strength and sensibility, to reasoning about that flavor of category in general?

In

"A Practical Type Theory for Symmetric Monoidal Categories"

Mike Shulman develops such a list of criteria and then follows them towards a new type theory for symmetric monoidal categories, PTT.

1.1 The Criteria

While Shulman provides a longer list for the specific case of symmetric monoidal categories, two items provide a more general principle. Any practical $\mathsf{Semantics} \dashv \mathsf{Syntax}$ relationship should enjoy:

1. an initiality theorem:
   (a) say something nice here
   (b) $\mathsf{Semantics}\left(\mathsf{Syntax}\left(\mathscr{C}\right)\right) \xrightarrow{\sim} \mathscr{C}$; and
2. possess term and typing judgements which permit us to reason about types as "sets with elements", albeit with some carefully chosen restrictions, e.g. coherence, linearity, etc.).

In this blog post we'll :

- try to convince you that these requirements, while simple to state, are far more sophisticated than they might appear;
- we'll follow them towards a syntax for symmetric monoidal categories, indeed we'll see precisely how these two requirements almost innevitably lead to Shulman's PTT as a synthesis of Sweedler's notation and Street&Batanin's Computads

[1] or, more broadly, some formal language
[2] As Shulman points out, a far more complete list can be found in Part D of "The Elephant"

- we'll explore a specific instance of Shulman's practical type theory, and see how these requirements play out;
- ; and more
- in this last point we'll suggest a little of the direction of the ACT project.

1.2 Initiality

What are we talking about when we talk about categories with type theory?

1.3 Sets With Elements

The great surge in interest in the relationship between type theory and category theory has, of course, been spurred by the advent of homotopy type theory. In particular, homotopy type theory has shown how particular principled choices in a syntax for elementary reasoning can be used to reason about structures which are far from discrete. [3]

—-THIS NEEDS FIXING—-

While the parametrization of HoTT by $(\infty, 1)$-toposes is present in the library of higher inductive types which are put into the theory as primitive rules; every $(\infty, 1)$-topos is locally presentable, meaning it is the

While HoTT thereby justifies our interest in such a relationship for symmetric monoidal categories, the character of such categories is in general, very different than the character of $(\infty, 1)$-toposes and indeed 1-toposes.

Typing judgements are.. such they should be maps

While $\times$ enjoys a universal property characterized in terms of projection maps, the universal property of $\otimes$ is characterized in terms of co-projection maps. In particular, for any $z \in X \times Y$, we have that

$$z = \mathsf{pair}\left(\mathsf{pr}_1\left(z\right), \mathsf{pr}_2\left(z\right)\right)$$

whereas no such characterization of the elements of $X \otimes Y$ is available. Locating the concern in the familiar terrain of vector spaces provides this categorical phenomenon with concrete intuition; it is the fact that the tensor product $\bigotimes_{i \in \{1,\ldots,n\}} V_i$ of vector spaces is comrised not of simple tensors

$$v = v_1 \otimes \cdots \otimes v_n = \mathsf{tuple}\left(v_1, \ldots, v_n\right)$$

but of linear combinations of them. Any *practical* type theory for symmetric monoidal categories

We can however, pretend that every tensor is a tuple, provided we are very careful; this is exactly the power of Sweedler's notation which Shulman's expands into a syntax.

1.4 Practical Type Theory

[3] It should of course be noted that synthetic geometry set the stage for such a PROGRAM as it demonstrates very well that a sophisticated treatment of elements can in fact

SymmMonCat	$\{s, t : \mathfrak{F}\mathcal{R} \rightrightarrows \mathfrak{F}\mathcal{G}\}$	PTT
Objects	$\left\{ (A_i)_{i \in \{1,\ldots,n\}} \middle\vert n \in \mathbf{N},\ A_i \in \mathsf{Ob}(\mathcal{R}) = \mathsf{Ob}(\mathcal{G}) \right\}$	Types
Morphisms	$\circ\&\otimes$-closure of $\mathsf{Mor}(\mathcal{G})$	Typing Judgements
Commuting Diagrams	$(s(f) = f' \wedge t(f) = f'') \Rightarrow f' \sim_{\mathcal{R}} f''$	Equality Judgments

Table 1: Cat and Type Theory