NOTES ON FIBRATIONAL SEMANTICS

DANIEL GRATZER

Abstract. I have lately found myself reading a large number of papers which deal in some way with fibrations in category theory. Particularly to give semantics to a logic or type theory. In order to be sure that I've internalized these concepts in some small way I have written up some notes introducing fibrations.

It would be far too ambitious to claim the completeness of these notes (this would be closer to a dissertation [1] rather than a note). Rather, I shall aim for a level of completeness that let's me state and prove semantics for an interesting type theory. After that, I will gesture towards how the semantics might be extended for more featureful logics or type theories but avoid the same level of proof.

1. Introduction

Let us first start with a brief discussion motivating categorical semantics in general and fibrations in particular. Denotational semantics is a rich and well studied area of type theory dating back to work by Scott and Strachey. The core idea is to reduce the studying of programs to the study of better understood mathematical objects. Originally these tended to be things like *sets* or *continuous lattices*. As time has evolved though we have shifted to objects such as *games* or other exotic objects. Lately however the field of categorical semantics has exploded. Categorical semantics in some sense acts as glue between syntax and traditional denotational semantics.

With categorical semantics one doesn't use concrete. Instead, categorical semantics aim to axiomatize the structure necessary for any concrete model to be to possess. The classic example is that of a cartesian closed category (CCC). This possesses enough structure to model precisely the simply typed lambda calculus. Now the power of categorical semantics becomes apparent, by proving that we can model the simply typed lambda calculus in any CCC, we have suddenly reduced the problem of finding models to the problem of finding CCCs. This is a comparatively easy and well-studied problem. In fact, this immediately gives us dozens of models of the simply typed lambda calculus for free

(1) **Set**

(2) **Dom**

(3) **Eq**

(4) **PEq**

(5) ...

Date: May 5, 2017.

This illustrates the role of categorical semantics. They mediate between the purely syntactic demands of a type theory and the mathematically natural constraints that we need to impose on a category to obtain a model. By finding a convenient set of constraints for a model, models become cheap and easy to find.

This is the role of categorical semantics but what about fibrational semantics? Fibrations are the categorical version of indexing. Indexing is a prevalent phenomenon in type theory; terms depend on types, types may depend on other types, types may even depend on terms! It turns out that be starting with a notion of indexing and imposing constraints on the indexed and the indexing categories we can quickly and flexibly model type theory. Again we encounter the advantages of generality become apparent when working with fibrational models. By specifying the properties we require of a fibration to provide in order to support a model of a type theory, we generalize over even the variety of fibrations that support this. For instance, if a split fibration yields a model of predicate logic then we are not merely limited to one source of fibrations. Throughout these notes the generality that this provides will make it apparent how fibrational semantics generalize the ideas of models into locally cartesian closed categories, toposes, and regular categories.

2. An Introduction to Fibrations

Before diving immediately into the categorical formulation of fibrations I would like to start with the set-theoretic intuition. In set theory, families of sets are a common occurrence. They are usually denoted $(X_i)_{i\in I}$ so that for any element $i \in I$ we have a designated set X_i. This presentation is what we shall refer to as the *index-based* presentation of families of sets. An equivalent and useful way to view this family is as a function

$$\iota : X = \bigcup_{i\in I} X_i \to I$$

we will refer to X as the *total* set and I as the *index* set. The idea is that every element $x \in X_i$ instead becomes $(i, x) \in X$, a pair of the index at which it lives and the actual element. The map ι, the so called *presentation* map, will then send $(i, x) \mapsto i$. We can then easily recover our original X_is from ι.

$$X_i = \iota^{-1}(i)$$

For a user of set theory this presentation may seem slightly stilted but it does have the major advantage that it's purely based on maps between sets. This is something we can conceivably generalize to categories since categories possess this structure. The next question what properties of presentation maps to we wish to preserve when we categorify it.

One feature that will turn out to be of great importance is the *reindexing* of families of sets. For sets, if we have a map $f : I \to J$ and $(X_i)_{i\in I}$ we can conceivably form a family of sets which is indexed by J, $(Y_j)_{j\in J}$, where Y_j is defined as

$$Y_i = X_{f(i)}$$

This reindexing again only depends on maps between sets so it is quite natural to categorify. The first way to categorify this is to maps sets to objects and functions as morphisms. We

then define a family as a morphism $f : B \to A$. We generally will write this as B *over* A or pictorially

$$\begin{array}{c} B \\ \downarrow{\scriptstyle f} \\ A \end{array}$$

We cannot naturally recover what B_a might mean yet because we do not have the same notion of elements that we do in set theory. Instead, we will see later that this is a general result of reindexing.

Reindexing reappears here quite naturally in finite limit categories in the form of pullbacks. Specifically, if we cast a set theoretic function $f : X \to Y$ as a morphism $f : I \to J$, given a family $b : A \to I$ we can define the reindexing $f^*(b)$ as the pullback

$$\begin{array}{ccc} f^*(A) & \longrightarrow & A \\ {\scriptstyle f^*(b)}\downarrow & & \downarrow{\scriptstyle b} \\ I & \xrightarrow[f]{} & J \end{array}$$

Now that we have generalized the notion of indexing from sets, we can start to observe some of the categorical structures which arise from this *indexing-as-presentations* approach. First, families over A for some given A quite naturally form a category: $\mathbb{C}/A$.

Definition 2.1. *The* slice category *over A, written $\mathbb{C}/A$ is defined as the category with*

Objects: *Objects are families $f : B \to A$*

Morphisms: *A morphism between $f : B \to A$ and $g : C \to A$ is a morphism of $h \in \hom_{\mathbb{C}}(B, C)$ so that $f = g \circ h$.*

$$\begin{array}{ccc} B & \xrightarrow{\quad h \quad} & B \\ & {\scriptstyle f}\searrow \quad \swarrow{\scriptstyle g} & \\ & A & \end{array}$$

Now that we have categories for families over A, it is natural to ask how the reindexing maps f^* act on these categories. In particular, do these respect the structure of slice categories and thus form a functor. As it turns out, this is the case.

Theorem 2.2. *For any $f : A \to B$ in a finite limit category, reindexing along f gives a functor $f^* : \mathbb{C}/B \to \mathbb{C}/A$.*

Proof. TODO □

This sort of structure, categories of indexed families with reindexing functors between them will turn out to be a natural setting for models of type theory. To give a flavoring for why this is interesting, we will model each $\vdash \Gamma\,\mathrm{ctx}$ as an object our category, each $\Gamma \vdash A\,\mathrm{type}$ as a family over $[\![\vdash \Gamma\,\mathrm{ctx}]\!]$ and terms as morphisms between these families. The reindexing functor correspond naturally to substitution. With this we reduce the problem of axiomatizing our

semantics to merely demanding the existence of certain categorical constructs in each slice category. In other words, we will use the categorified indexing to represent how types and terms depend on contexts. By abstracting away this structure now then, we can easily wipe away half the work of defining categorical semantics.

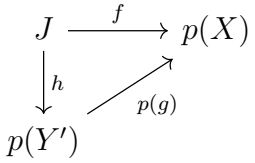
In order to generalize away from one particular form on indexing, we isolate the idea of having a having a family of categories varying over a base or index category. This is a fibration. I will start by giving the formal definition of a fibration which makes references to some as of yet undefined concepts. I will then progressively attempt to fill it in so that the original, formal, definition becomes more intuitive.

Definition 2.3. *A fibration, $p : \mathbb{E} \to \mathbb{B}$, so that for every morphism $u \in \hom_{\mathbb{B}}(I, p(Y))$ there is a cartesian lifting $\bar{u} : u^*(Y) \to Y$.*

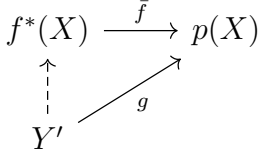
Intuitively a fibration is a way of gluing a collection of categories indexed by another category so we start with a *display functor* $p : \mathbb{E} \to \mathbb{B}$. We will think of $\mathbb{E}$ as the collection of all the objects and morphisms of our indexed categories clumped together and p tells us the index of some particular object. Accordingly, we can then say that a morphism, $f \in \hom_{\mathbb{E}}(A, B)$ is vertical if $p(f) = 1$. It is obvious that $\mathbb{E}_I$ comprised of objects X so that $p(X) = I$ and morphisms vertical over I forms a category. It is important that $\mathbb{E}$ is not necessarily just $\amalg_I \mathbb{E}_I$, the morphisms that are not vertical will important for relating the structure $\mathbb{B}$ to $\mathbb{E}_I$.

What we would like next is a way from moving between $\mathbb{E}_I$ and $\mathbb{E}_J$ using the information we have from $\hom_{\mathbb{B}}(J, I)$. This is just a generalization of what we did with slice categories to fibrations. In order to do this, given a map $f : J \to I$ we wish to find a way to transport from $\mathbb{E}_I$ to $\mathbb{E}_J$. In order to do this, we wish to lift the morphism in some way so that given X so that $p(X) = I$ we have a map $X \to Y$ for some Y so that $p(Y) = J$ that projects down onto f. If we think of this as substitution over contexts (like our above example) this roughly corresponds to saying that if we have a way to chance our context to a different context, we can change types in the first context to types in the second. Something that seems quite plausible, it ought to be just substitution on types.

It would be ideal if this morphism was canonical in some sense. It's quite easy to imagine a scenario where there are infinitely many different morphisms that we could lift to. If we think of this operation as corresponding to substitution, then there must be a canonical choice for us to pick. Now for some notation, we write this hypothetical *best substitution* as a morphism in $\mathbb{E}$, $\bar{f} : f^*(X) \to X$ in order to evoke the similarity to the pullback case we started with. As is usual in category theory, we will characterize this lifting using a universal property. Namely, we want this morphism to be terminal. This means that given



we want a canonical map



in this case, our lifting $\bar{u}$ is said to be *cartesian*.

Definition 2.4. *A map $f : X \to Y$ is said to be cartesian over $u : f(Y) \to f(X)$ if $p(f) = u$ and if for any map $g : Y' \to X$ so that $u \circ v = p(g)$, there is a unique $h : Y' \to Y$ so that $p(h) = v$ and $f \circ h = g$.*

With this we have at least acquired the necessary technical information to understand what a fibration is, however we're still lacking in some intuition. In particular, the connection between cartesian maps and reindexing functors remains unclear. What we really wanted was a family of functors u^* for each $u : J \to I$ mapping $\mathbb{E}_I$ to $\mathbb{E}_J$. Well as it happens, this functor is definable using cartesian maps.

Theorem 2.5. *For a fibration $p : \mathbb{E} \to \mathbb{B}$, any map $u : J \to I$ in $\mathbb{B}$ induces a functor $u^* : \mathbb{E}_I \to \mathbb{E}_J$.*

Proof. todo □

In order to see that this characterization is even equivalent to this sort of indexing structure, we will next investigate the so called *Grothendieck construction*. This allows us to turn a pseudofunctor $\mathbb{B} \to \mathbf{Cat}$ into a particular fibration. This lets us know that we haven't lost or gained anything essential by switching to the fibrational presentation of indexing.

Theorem 2.6. *Every pseudofunctor $\mathbb{B} \to \mathbf{Cat}$ corresponds to a fibration over $\mathbb{B}$.*

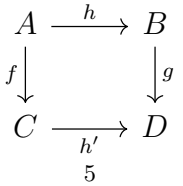
Proof. todo □

Having set out this very abstract definition, I will now relate it to what we were previously studying: indexed families of sets. In particular, let us look at the so called *codomain fibration*: Cod. In order to do this, let us start with a few preliminary definitions.

Definition 2.7. *The arrow category over a category, $\mathbb{C}$, is written $\mathbb{C}^{\to}$ and is defined with*

Objects: *For objects arrows $f : A \to B$*

Morphisms: *A morphism between $f : A \to B$ and $g : C \to D$ is a pair of morphisms (h, h') so that $h : A \to C$ and $h' : B \to D$ and the following commutes*



The reader is referred to Mac Lane [3] for a more detailed investigation of the arrow category. For our purposes we are largely interested in the codomain functor $\mathrm{Cod} : \mathbb{C}^{\to} \to \mathbb{C}$. It is defined

$$\mathrm{Cod}(f : A \to B) = B$$
$$\mathrm{Cod}(h, h') = h'$$

it is trivial to check this indeed defines a functor. More than this though, in a category with pullbacks this functor is in fact a fibration.

Example 2.8. *The* Cod *functor defines a fibration.*

Proof. todo □

Now an important step of understanding a fibration $p : \mathbb{E} \to \mathbb{B}$ is to study the fibers that it defines, $\mathbb{E}_I$. In particular, we shall find this illuminating in the case of the codomain fibration

Lemma 2.9. *The fibers of* $\mathrm{Cod} : \mathbb{C}^{\to} \to \mathbb{C}$ *are precisely the slice categories of* $\mathbb{C}$*. Moreover, reindexing in the fibration corresponds to the pullback functor of slice categories.*

Proof. todo □

Have done this we have come full circle: our very abstract notion of fibration can be specialized to simple set theoretic indexing by instantiating a particular fibration on a particular category.

Having shown that our abstraction is a useful one, we can develop the theory of fibrations abstractly. The reader is referred to Jacobs [2] for a more in-depth coverage of fibered category theory. For our purposes, I only want to flesh out enough theory to explain the semantics of System F in the next section.

3. Semantics for System F

4. Extending Fibrational Semantics to Other Theories

5. Fibrational Semantics Compared to Other Approaches

6. Conclusion

References

[1] B. Jacobs. *Categorical type theory.* PhD thesis, University of Nijmegen, 1991.

[2] B. Jacobs. *Categorical Logic and Type Theory.* Studies in logic and the foundations of mathematics. Elsevier, 2001.

[3] Saunders Mac Lane. *Categories for the Working Mathematician.* Graduate Texts in Mathematics. Springer New York, 1998.