Categorical Semantics of Type Theories

Harley D. Eades III
Computer Science
The University of Iowa

May 1, 2013

Abstract

Category theory and its applications to type theory are well known and have been explored extensively. In this manuscript we present basic category theory and give a categorical semantics to a large class of type theories. In this document we emphasize rigor and give as much detail as possible with respect to the abilities of the authors. This document will also be self contained.

Contents

0.1 Introduction 2

1 Category Theory **3**
1.0.1 Initial Algebras 3
1.0.2 Final Coalgebras 3

2 Simple Type Theories **5**
2.1 A Metaframework 5
2.2 The Theory of Constants 5

3 Data Types **7**
3.1 Coalgebras and Analysis 7
3.1.1 Real Numbers 7

List of Figures

2.1 Syntax of the theory of constants 5
2.2 Type assignment for the theory of constants 6
2.3 Definitional equality for the theory of constants 6

0.1 Introduction

We give an amazingly rich and engaging introduction.

Chapter 1

Category Theory

1.0.1 Initial Algebras

1.0.2 Final Coalgebras

Lets consider the infinite stream of ones.

$$1 :: 1 :: 1 :: 1 \cdots$$

Now what can be said about such an object? We definitely can make observations about it. That is, we can observe that, say, the first element is a 1. In addition to that we can observe that the second element is also a 1. Furthermore, we can make the observation that what follows the second element is still an infinite streams of ones. We can define this stream using a few morphisms. That is $\mathsf{head} : \mathcal{A} \to \mathbb{N}$ and $\mathsf{next} : \mathcal{A} \to \mathcal{A}$ are the morphisms, where $\mathcal{A}$ is some fixed set. Now notice that for any $a \in \mathcal{A}$ we have $\mathsf{head}(a) = 1$, and $\mathsf{next}(a) = a' \in A$. Now if we compose these two we obtain $\mathsf{head}(\mathsf{next}(a)) = \mathsf{head}(a') = 1$. Thus, using these morphisms we can completely define our stream above by taking for each position $n \in \mathbb{N}$ in the stream $\mathsf{head}(\mathsf{next}^n(a))$ to be the value at that position for any $a \in \mathcal{A}$. These two morphisms head and next are the observations we can make about the element a. It just happens, in this simple example, that the observations are the same for all elements of $\mathcal{A}$.

At this point we can take the product of the two morphisms head and next to obtain the morphism $\langle \mathsf{head}, \mathsf{next} \rangle : \mathcal{A} \to \mathbb{N} \times \mathcal{A}$. If we define the functor $\mathcal{F}(X) := \mathbb{N} \times X$ then we can redefine the product as $\langle \mathsf{head}, \mathsf{next} \rangle : \mathcal{A} \to \mathcal{F}(\mathcal{A})$. Furthermore, if we make $\mathbb{N}$ arbitrary and call it say $\mathcal{B}$ then we obtain the functor $\mathcal{F}(X) := \mathcal{B} \times X$. Using this new definition of the functor $\mathcal{F}$ and $\langle \mathsf{head}, \mathsf{next} \rangle : \mathcal{A} \to \mathcal{F}(\mathcal{A})$ we can define any infinite stream of elements of $\mathcal{B}$. Taking the functor $\mathcal{F}$ and the pair $(\mathcal{A}, \langle \mathsf{head}, \mathsf{next} \rangle)$ we obtain what is called a coalgebra.

Definition 1 (Coalgebra)**.** *A **coalgebra** or **$\mathcal{F}$-coalgebra** is a pair $(\mathcal{F}, s_{\mathcal{A}})$ with respect to some category $\mathcal{C}$ consisting of an object $\mathcal{A}$ called the carrier object of the coalgebra, an endofunctor $\mathcal{F} : \mathcal{C} \to \mathcal{C}$, and a morphism $s_{\mathcal{A}} : \mathcal{A} \to \mathcal{F}(\mathcal{A})$ called the structure of the coalgebra.*

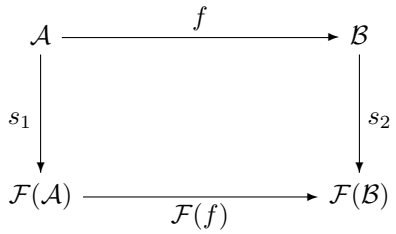
It is often the case that coalgebras are denoted simply by specifying the type of the structure of the coalgebra. That is $s_{\mathcal{A}} : \mathcal{A} \to \mathcal{F}(\mathcal{A})$. This is precise, because it gives all of the elements of a coalgebra, but the category, but this is usually determinable from the context. So, in our example above we can see that $\langle \mathsf{head}, \mathsf{next} \rangle : \mathcal{A} \to \mathcal{F}(\mathcal{A})$ is indeed a coalgebra with respect to the category Set.

Now consider the infinite stream of twos.

$$2 :: 2 :: 2 :: 2 \cdots$$

This is definable by a coalgebra $\langle \hat{\mathsf{head}}, \hat{\mathsf{next}} \rangle : \mathcal{B} \to \mathcal{F}(\mathcal{B})$ with respect to the category Set. Suppose further that there is a function $f : \mathcal{A} \to \mathcal{B}$ such that $\hat{\mathsf{head}} \circ f = \mathsf{head}$ and $\hat{\mathsf{next}} \circ f = f \circ \mathsf{next}$. In this case f is called a homomorphism between coalgebras.

Definition 2 (Homomorphism)**.** *A **homomorphism** between two coalgebras $s_1 : \mathcal{A} \to \mathcal{F}(\mathcal{A})$ and $s_2 : \mathcal{B} \to \mathcal{F}(\mathcal{B})$ is a morphism $f : \mathcal{A} \to \mathcal{B}$ such that the following diagram commutes:*



The above diagram expresses that f commutes with the operations of the two coalgebras.

Definition 3 (Final Coalgebras)**.** *A **final coalgebra** $s_1 : A \to \mathcal{F}(\mathcal{A})$ is one such that for any other coalgebra $s_2 : \mathcal{B} \to \mathcal{F}(\mathcal{B})$ there is a unique homomorphism between them.*

Chapter 2

Simple Type Theories

2.1 A Metaframework

We will use Martin-Löf's Type Theory for our metaframework throughout this chapter. This will allow use to give precise types to all of the structures in our object languages. In fact all of the type theories discussed in this chapter can be rigorously defined within this framework where binding can be encoded using either de Bruijn indecies or using the locally nameless representation [1] [**?**].

2.2 The Theory of Constants

We begin our journey into the world of categorical semantics of type theories by first showing how to interpret a simple algebraic theory consisting of a countably infinite set of variables, a finite set of constant types, a finite set of i-ary function symbols, a typing judgment, and a definitional equality judgment. This theory is called the theory of constants. We first give a formal definition of this theory in the presentation we will adopt for the reminder of this document. The syntax for the theory of constants is defined in Figure 2.1.

The free variables of the theory can be defined at the metalevel as de Bruijn indices, but we will use mathematical notation to simplify the presentation. They have type Term and are classified by constant types of type Type. The i-ary function symbols have type $\mathbf{Term}^i \Rightarrow \mathbf{Term}$. The constant types of the theory of constants have meta-type Type. Judgments are metastatements describing what type a term can be assigned. All of the judgements we will define can be defined at the metalevel as an inductive datatype where each rule of the judgment is defined as a constructor. If we call the typing judgment has_type at the type level then its type is $(\Gamma : [\mathbf{Term} \times \mathbf{Type}]) \Rightarrow (t : \mathbf{Term}) \Rightarrow (U : \mathbf{Type}) \Rightarrow \mathbf{Type}$. We denote this judgment by $\Gamma \vdash t : U$. The type of the definitional equality judgment is similar. We define the type assignment judgment in Figure 2.2 and the definitional equality judgment in Figure 2.3.

[1] We prefer the latter.

$$
\begin{array}{rrcl}
(Types) & T & ::= & S \mid U \\
(Terms) & t & ::= & x \mid c \mid f\ x_1 \ldots x_i \\
(Contexts) & \Gamma & ::= & x : T \mid \Gamma_1, \Gamma_2
\end{array}
$$

Figure 2.1: Syntax of the theory of constants

$$\frac{}{\Gamma, x : S, \Gamma' \vdash x : S}\ \textsc{Var} \qquad \frac{}{x_1 : S_1,\ \ldots, x_i : S_i \vdash f\ x_1 \ldots x_i : U}\ \textsc{Fun}$$

$$\frac{}{\Gamma \vdash c : U}\ \textsc{Const}$$

Figure 2.2: Type assignment for the theory of constants

$$\frac{}{\Gamma \vdash t = t : T}\ \textsc{Refl} \qquad \frac{\Gamma \vdash t_1 = t_2 : T}{\Gamma \vdash t_2 = t_1 : T}\ \textsc{Sym}$$

$$\frac{\Gamma \vdash t_1 = t_2 : T \quad \Gamma \vdash t_2 = t_3 : T}{\Gamma \vdash t_1 = t_3 : T}\ \textsc{Trans} \qquad \frac{\Gamma \vdash t_1 = t_2 : T_1 \quad \Gamma, x : T_1 \vdash t = t' : T_2}{\Gamma \vdash [t_1/x]t = [t_2/x]t' : T_2}\ \textsc{Subst}$$

Figure 2.3: Definitional equality for the theory of constants

Chapter 3

Data Types

3.1 Coalgebras and Analysis

It is widely known that the natural numbers and induction come hand in hand. That is induction is used to both define and prove predicates on the natural numbers. That is elementary arithmetic. So we have the following picture:

$$\frac{\text{Induction}}{\text{arithmetic (Natural Numbers)}}$$

We then dare pose the question, how do we fill in the hole in the following picture?

$$\frac{\text{Coinduction}}{?}$$

Well D. Pavlović, V. Pratt, and M. Escardo have come up with an answer for us. They answer with the following:

$$\frac{\text{Coinduction}}{\text{analysis (real numbers)}}$$

In this section we will slowly uncover how this works by closely examining their work. One thing to notice regarding the relationship between the above pictures is that induction and coinduction are duels in a very precise way. We pose yet another question, does this imply that arithmetic and analysis are duels? Are natural numbers and real numbers duals? These are unknown questions as of right now.

3.1.1 Real Numbers