Categorical Semantics of Type Theories

Harley D. Eades III
Computer Science
The University of Iowa

January 16, 2014

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

List of Figures

0.1 Introduction

We give an amazingly rich and engaging introduction.

Chapter 1

Category Theory

1.1 Natural Transformations

Lemma 1 (Composing Natural Transformations)**.** *Suppose $\phi : F_1 \to F_2$ and $\psi : F_2 \to F_3$ are natural transformations between the functors F_1, F_2, and F_3. Then $\psi \circ \phi : F_1 \to F_3$ is a natural transformation.*

Proof. Suppose $\phi : F_1 \to F_2$ and $\psi : F_2 \to F_3$ are natural transformations between the functors F_1, F_2, and F_3. Then we know the following by the definition of a natural transformation:

$$\begin{array}{ccc} a & F_1(a) \xrightarrow{\phi_a} & F_2(a) \\ \Big\downarrow{\scriptstyle f} & \Big\downarrow{\scriptstyle F_1(f)} & \Big\downarrow{\scriptstyle F_2(f)} \\ b & F_1(b) \xrightarrow[\phi_b]{} & F_2(b) \end{array}$$

and

$$\begin{array}{ccc} a' & F_2(a') \xrightarrow{\psi_{a'}} & F_3(a') \\ \Big\downarrow{\scriptstyle g} & \Big\downarrow{\scriptstyle F_2(g)} & \Big\downarrow{\scriptstyle F_3(g)} \\ b' & F_2(b') \xrightarrow[\psi_{b'}]{} & F_3(b') \end{array}$$

where all of the above diagrams commute. Now it suffices to show that they following commutes:

$$\begin{array}{ccc} c & F_1(c) \xrightarrow{\psi_c \circ \phi_c} & F_3(c) \\ \Big\downarrow{\scriptstyle h} & \Big\downarrow{\scriptstyle F_1(h)} & \Big\downarrow{\scriptstyle F_3(h)} \\ d & F_1(d) \xrightarrow[\psi_d \circ \phi_d]{} & F_3(d) \end{array}$$

The fact that the previous diagrams commute follows from the following equational reasoning:

$$\begin{aligned}(\psi_d \circ \phi_d) \circ F_1(h) &= \psi_d \circ (\phi_d \circ F_1(h)) \\ &= \psi_d \circ (F_2(h) \circ \phi_C) \\ &= (\psi_d \circ F_2(h)) \circ \phi_C \\ &= (F_3(h) \circ \psi_C) \circ \phi_C \\ &= F_3(h) \circ (\psi_C \circ \phi_C)\end{aligned}$$

Therefore, the composition of two natural transformations is also a natural transformation. □

1.2 Monoidal Categories

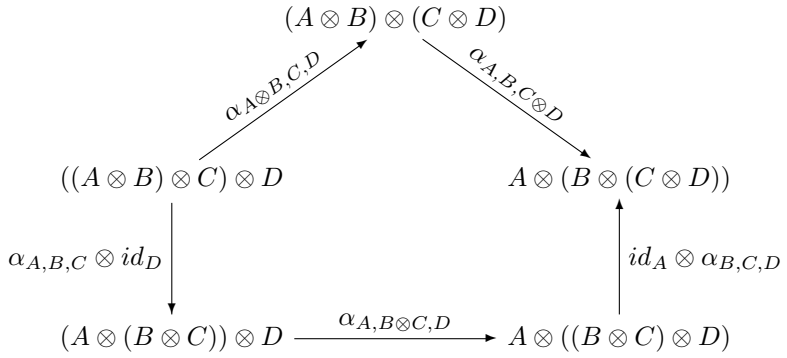
In this section we develope the notions of monoidal categories and symmetric monoidal categories. These types of categories are used extensively in the study of linear logic and linear type theories. We first define each of these types of categories and then prove the well-known coherence property. The following defines monoidal categories.

Definition 2. *A **monoidal category** is a tuple $(C, \otimes, T, \alpha_{A,B,C}, \lambda_A, \rho_A)$, where C is a category, $\otimes$ is called the multiplication, $T \in C_0$ is the unit, $\alpha_{A,B,C}$ is a natural isomorphism called the associator, λ_A is a natural isomorphism called the left unitor, and ρ_A is a natural isomorphism called the right unitor. The components of the monoidal category have the following types:*

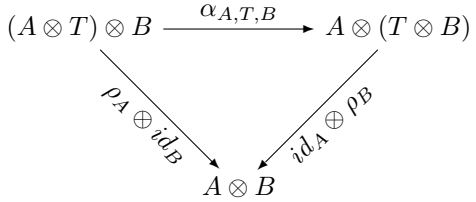
$$C \times C \xrightarrow{\otimes} C \qquad (A \otimes B) \otimes C \xrightarrow{\alpha_{A,B,C}} A \otimes (B \otimes C)$$

$$T \otimes A \xrightarrow{\lambda_A} A \qquad A \otimes T \xrightarrow{\rho_A} A$$

Monoidal categories have the following coherence conditions:



and



The first coherence condition is known as the pentagon identity and the second the triangle identity. We also need the following equality to hold:

$$\lambda_T = \rho_T : T \otimes T \to T$$

Now a symmetric monoidal category is one in which the monoidal multiplcation is symmetric.

Definition 3. *A **symmetric monoidal category (SMC)** is a monoidal category with an additional natural isomorphism:*

$$A \otimes B \xrightarrow{\beta_{A,B}} B \otimes A$$

Satisfying the following coherence conditions:

$$\begin{array}{ccccc} (A \otimes B) \otimes C & \xrightarrow{\alpha_{A,B,C}} & A \otimes (B \otimes C) & \xrightarrow{\beta_{A,B\otimes C}} & (B \otimes C) \otimes A \\ \downarrow{\scriptstyle \beta_{A,B} \otimes id_C} & & & & \downarrow{\scriptstyle \alpha_{B,C,A}} \\ (B \otimes A) \otimes C & \xrightarrow{\alpha_{B,A,C}} & B \otimes (A \otimes C) & \xrightarrow{id_B \otimes \beta_{A,C}} & B \otimes (C \otimes A) \end{array}$$

and

$$\beta_{B,A} \circ \beta_{A,B} = id_{A\otimes B}$$

1.2.1 Free Symmetric Monoidal Categories

First, we define free monoidal categories in the style of Mac Lane in [**?**]. The language of **binary words** is defined by the following grammar:

$$\text{(binary words)} \quad w, v \quad ::= \quad \mathrm{e}_0 \mid - \mid w \,\square\, v$$

The length of a binary word is defined by recursion in the following way:

$$\begin{array}{lcl} length(\mathrm{e}_0) & = & 0 \\ length(-) & = & 1 \\ length(w \,\square\, v) & = & length(w) + length(v) \end{array}$$

An example of a binary word of length 4 is $(- \,\square\, -) \,\square\, (- \,\square\, (- \,\square\, \mathrm{e}_0))$.

The cateogry W of binary words is defined by taking for any two binary words v and w such that $length(v) = length(w)$ exactly one arrow $v \to w$. Certainly, this definition includes identity arrows $id_w : v \to v$. Composition also clearly exists, because having two arrows $f : w \to w'$ and $g : w' \to v$ means that the length of the words w,w', and v all coincide, thus, the morphism $g \circ f : w \to v$ must exist. Identities are certainly unique by construction. Thus, identitiy axioms hold. The only thing left to verify is associativity. There is exactly one arrow between any two objects, thus, it must be the case that $(g \circ f) \circ j = g \circ (f \circ j)$.

It is easy to see that the previous construction makes W a preorder. Furthermore, notice that all arrows are invertible. We know for any two objects w and v there is an arrow $f : w \to v$, but we also know there is an arrow $f^{-1} : v \to w$ by construction. In addition we know that there is exactly one arrow between any two objects, thus, their composition must be the identitiy. Keeping this in mind it is easy to see that W is also a monoidal category with the multiplcation being $w, v \mapsto w \,\square\, v$, e_0 being the unit, and the isomorphisms λ, ρ, and α defined to be their respective arrows in W. The coherence conditions for monoidal categories are guaranteed to hold by the fact that every diagram in W is commutative by construction. Therefore, we have arrived at the following result.

Theorem 4 (The Free Monoidal Category)**.** *For any monoidal category B and any object $b \in B_0$, there is a unique functor $W \to B$ with $- \mapsto b$.*

Proof. We can think of the objects of W has objects of B with "holes". The holes are indicated by the $-$ word. So to construct an object of B from an object W we have to "fill in these holes". For some $b \in B_0$ we write the desired functor as $w \mapsto w_b$, denoted $(w)_b$. We write w_b to suggest that we are filling the holes in w with the object b. Now on objects of W we define the desired functor by:

$$(\mathrm{e}_0)_b = T \quad (-)_b = b \quad (w \,\square\, v)_b = (w)_b \,\square\, (v)_b$$

We now prove that this functor uniquely determines every object $(w)_b \in B$. Suppose $(w)_b = (v)_b$ for some words w and v, and we must show that $w = v$. Clearly, w and v must have the same length n. So we proceed by induction on n. Suppose $n = 0$, then $(w)_b = (v)_b = (\mathsf{e}_0)_b = T$, thus, $w = v = \mathsf{e}_0$. Now suppose $n = 1$, this case is similiar to the previous. Finally, suppose $n = m > 1$, then $w = w_1 \,\square\, w_2$ and $v = v_1 \,\square\, v_2$. Then

$$\begin{array}{rcl}(w_1 \,\square\, w_2)_b & = & (v_1 \,\square\, v_2)_b \\ & = & (w_1)_b \,\square\, (w_2)_b \\ & = & (v_1)_b \,\square\, (v_2)_b.\end{array}$$

Hence it must be the case that $(w_1)_b = (v_1)_b$ and $(w_2)_b = (v_2)_b$. By the induction hypthesis $w_1 = v_1$ and $w_2 = v_2$. Therefore, $w_1 \,\square\, w_2 = v_1 \,\square\, v_2$.

Next we prove the functor $(\,)_b$ is surjective. Suppose $b, c \in B_0$. We proceed by induction on the length c. If $c = T$, then $(\mathsf{e}_0)_b = T = c$. Now suppose $c = b$, then $(-)_c = c = b$. Lastly, suppose $c = c_1 \,\square\, c_2$, we know by induction there exsits two binary words w_1 and w_2 such that $(w_1)_b = c_1$ and $(w_2)_b = c_2$. Therefore, $(w_1 \,\square\, w_2)_b = c_1 \,\square\, c_2$, and we have shown that we have a bijection from W to B. □

1.3 Comma Categories

Definition 5. *Suppose $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{E}$ are categories. Given functors $F : \mathcal{E} \to \mathcal{C}$ and $G : \mathcal{D} \to \mathcal{C}$ the **comma category** $(F \downarrow G)$ is defined as follows:*

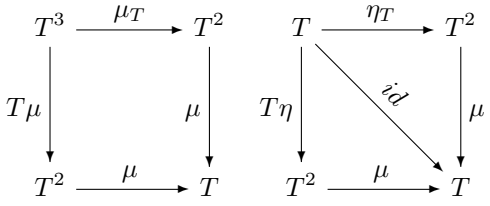
$$\frac{\begin{array}{l}e \in \mathcal{E}_0 \\ d \in \mathcal{D}_0 \qquad f : F\,e \to G\,d \in \mathcal{C}_1\end{array}}{(e, d, f) \in (F \downarrow G)_0}\ \textsc{Obj} \qquad \frac{\begin{array}{l}h : e \to e' \in \mathcal{E}_1 \qquad (e, d, f) \in (F \downarrow G)_0 \\ j : d \to d' \in \mathcal{D}_1 \qquad (e', d', f') \in (F \downarrow G)_0 \qquad f; G\,j = F\,h; f'\end{array}}{(h, j) : (e, d, f) \to (e', d', f') \in (F \downarrow G)_1}\ \textsc{Morph}$$

$$\frac{(e, d, f) \in (F \downarrow G)_0}{(\mathsf{id}_e, \mathsf{id}_d) : (e, d, f) \to (e, d, f) \in (F \downarrow G)_1}\ \textsc{Iden} \qquad \frac{\begin{array}{l}(h, j) : (e_1, d_1, f_1) \to (e_2, d_2, f_2) \in (F \downarrow G)_1 \\ (h', j') : (e_2, d_2, f_2) \to (e_3, d_3, f_3) \in (F \downarrow G)_1\end{array}}{(h; h', j; j') : (e_1, d_1, f_1) \to (e_3, d_3, f_3) \in (F \downarrow G)_1}\ \textsc{Comp}$$

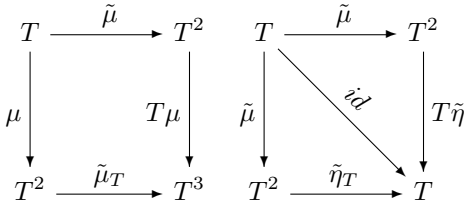
1.4 Monads and Comonads

In the sequel T^n, for $n \in \mathbb{N}$, is defined to be the n-fold composition of T, e.g. $T^2 = T \circ T$.

Definition 6. *A **monad** of a category C consists of an endofunctor $T : C \to C$ and two natural transformations $\eta : id_C \to T$ and $\mu : T^2 \to T$ satisfying the following commutative diagrams:*



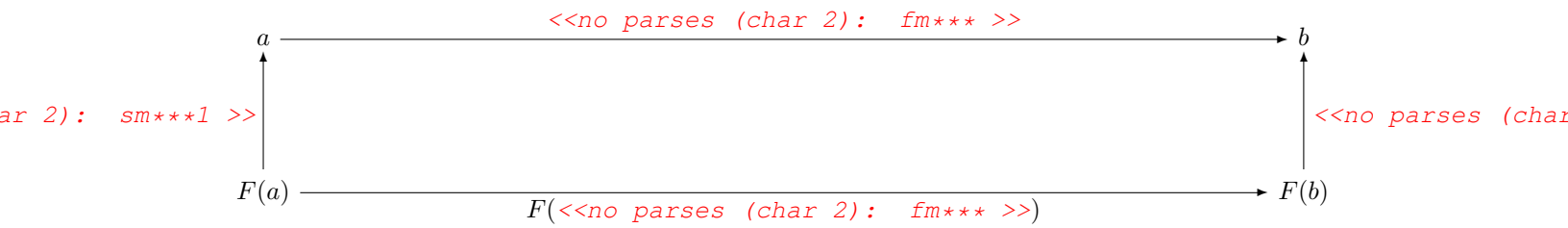
Definition 7. *A **comonad** of a category C consists of an endofunctor $T : C \to C$ and two natural transformations $\tilde{\eta} : T \to id_C$ and $\tilde{\mu} : T \to T^2$ satisfying the following commutative diagrams:*



1.5 Initial Algebras

Definition 8 (Algebra)**.** *An **algebra** or F**-algebra** is a pair $(F, \texttt{<<no parses (char 2): sm*** >>}_a)$ with respect to some category C consisting of an object a called the carrier object of the algebra, an endofunctor $F : C \to C$, and a morphism $\texttt{<<no parses (char 2): sm*** >>}_a : F(a) \to a$ called the structure of the algebra.*

Definition 9 (Homomorphism)**.** *A **homomorphism** between two algebras $\texttt{<<no parses (char 2): sm***1 >>} : F(a) \to a$ and $\texttt{<<no parses (char 2): sm***2 >>} : F(b) \to b$ is a morphism $\texttt{<<no parses (char 2): fm*** >>} : a \to b$ such that the following diagram commutes:*



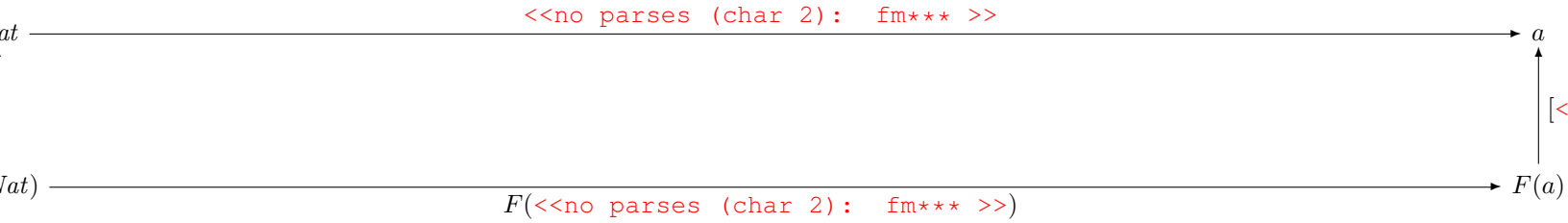
The above diagram expresses that $\texttt{<<no parses (char 2): fm*** >>}$ commutes with the operations of the two algebras.

Definition 10 (Initial Algebras)**.** *A **initial algebra** $\texttt{<<no parses (char 2): sm***1 >>} : F(a) \to a$ is one such that for any other algebra $\texttt{<<no parses (char 2): sm***2 >>} : F(b) \to b$ there is a unique homomorphism between them.*

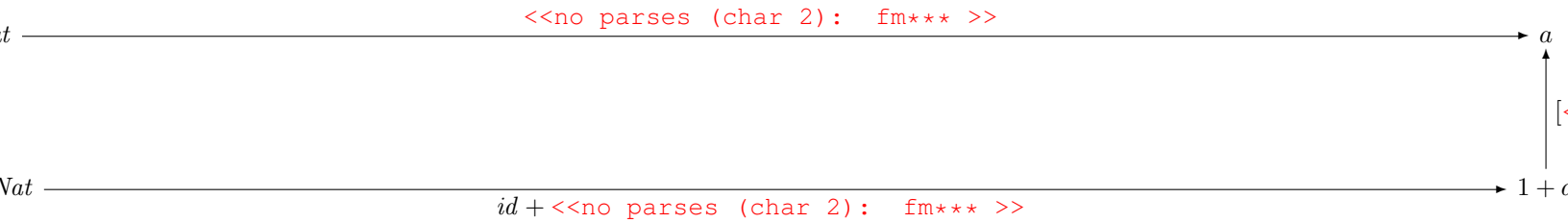
Lets consider a simple example. Suppose C is some category with an object Nat, and morphisms $\texttt{<<no parses (char 1): z}$ $1 \to Nat$ and $\texttt{<<no parses (char 2): sm*** >>} : Nat \to Nat$. Then it is possible to construct any natural number i using just these two morphisms. That is i is defined by $\texttt{<<no parses (char 2): sm*** >>}^i(\texttt{<<no parses (c}$ hence $\texttt{<<no parses (char 1): z***m >>}$ is the natural number zero and $\texttt{<<no parses (char 2): sm*** >>}$ is the successor operator. Now coparing these two operations up into a single morphism results in $[\texttt{<<no parses (char 1): z**}$ $1+Nat \to Nat$. Defining the functor $F(x) := 1+x$ we then obtain $[\texttt{<<no parses (char 1): z***m >>}, \texttt{<<no parses (c}$ $F(Nat) \to Nat$. Clearly, this is an F-algebra. Next we show that the algebra $[\texttt{<<no parses (char 1): z***m >>}, \texttt{<<no pa}$ $F(Nat) \to Nat$ is initial.

Lemma 11 (Initiality of the Natural Number Algebra)**.** *Let $F(x) := 1+x$. Then algebra $[\texttt{<<no parses (char 1): z***m >>},$ $F(Nat) \to Nat$ is initial.*

Proof. Suppose $F(x) := 1+x$ and $[\texttt{<<no parses (char 1): z***m' >>}, \texttt{<<no parses (char 2): sm***' >>}] :$ $F(a) \to a$ is an arbitrary algebra. It suffices to show that there exists a unique morphism $\texttt{<<no parses (char 2): fm*** >>} :$ $Nat \to a$ such that the following diagram commutes:



This diagram is equivalent to the following:



We know $F(\texttt{<<no parses (char 2): fm*** >>}) := id + \texttt{<<no parses (char 2): fm*** >>}$,
because the only morphisms from 1 to 1 are the identity morphisms and they are unique.
Take $\texttt{<<no parses (char 2): fm*** >>}(n) := \texttt{<<no parses (char 2): sm***' >>}^{(n)}(\texttt{<<no parses (cha}$
Where (n) is a inductively defined by the following equations:

$$\begin{array}{rcl}(\texttt{<<no parses (char 1): z***m >>}(*)) & := & 0 \\ (\texttt{<<no parses (char 2): sm*** >>}(a)) & := & 1 + (a).\end{array}$$

At this point we can see that:

$$\begin{array}{rcl}(\texttt{<<no parses (char 2): fm*** >>} \circ \texttt{<<no parses (char 1): z***m >>})(*) & = & \texttt{<<no parses (char} \\ & = & \texttt{<<no parses (char} \\ & = & \texttt{<<no parses (char} \\ & = & \texttt{<<no parses (char} \\ & = & (\texttt{<<no parses (cha}\end{array}$$

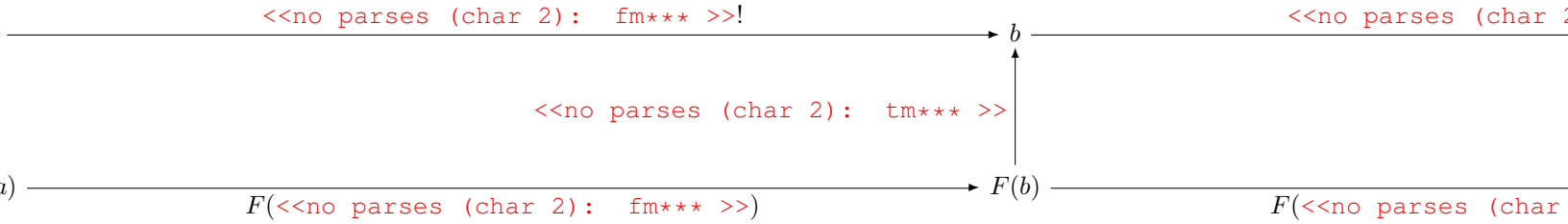
Thus,

$$\texttt{<<no parses (char 2): fm*** >>} \circ [\texttt{<<no parses (char 1): z***m >>}, \texttt{<<no parses (char 2): sm}$$

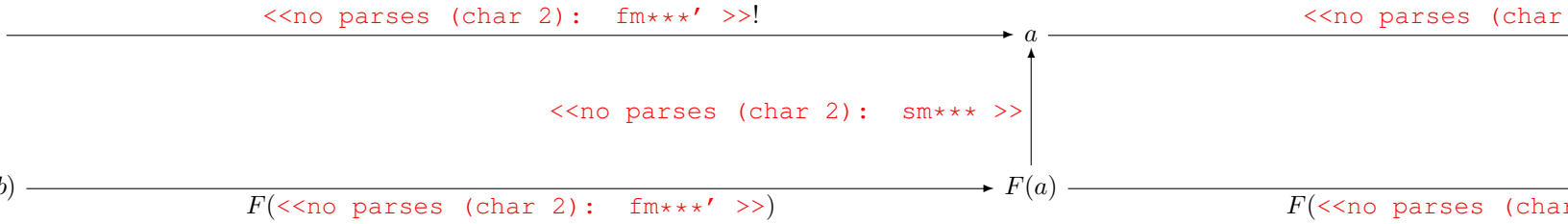
The previous results shows that the above diagram commutes. However, we still must show that $\texttt{<<no parses (char 2): fm***}$ is unique in order for it to be a homomorphism of algebras. Suppose $\texttt{<<no parses (char 2): gm*** >>}$: $Nat \to A$ such that $\texttt{<<no parses (char 2): gm*** >>} \circ [\texttt{<<no parses (char 1): z***m >>}, \texttt{<<no parses (}$ $[\texttt{<<no parses (char 1): z***m' >>}, \texttt{<<no parses (char 2): sm***' >>}] \circ (id + \texttt{<<no parses (char 2)}$ Then it must be the case that $\texttt{<<no parses (char 2): gm*** >>}(z(*)) = \texttt{<<no parses (char 1): z***m' >>}(*)$ and $\texttt{<<no parses (char 2): gm*** >>}(\texttt{<<no parses (char 2): sm*** >>}(a)) = id + \texttt{<<no parses (char}$ which can be easily shown to be equivalent to $\texttt{<<no parses (char 2): fm*** >>}$ by structural induction on the input to $\texttt{<<no parses (char 2): fm*** >>}$ and $\texttt{<<no parses (char 2): gm*** >>}$. Therefore, $\texttt{<<no parses (char 2): fm*** >>}$ is a homomorphism of algebras, and the algebra $[\texttt{<<no parses (char 1): z}$ $F(Nat) \to Nat$ for functor $F(x) := 1 + x$ is initial. □

Lemma 12 (Uniqueness of Initial Algebras)**.** *If* $\texttt{<<no parses (char 2): sm*** >>} : F(a) \to a$ *is an initial algebra of the functor* F*, then* $\texttt{<<no parses (char 2): sm*** >>}$ *is unique up to isomorphism.*

Proof. Suppose $\texttt{<<no parses (char 2): sm*** >>} : F(a) \to a$ is an initial algebra for the functor F. It suffices to show that the mediating homomorphism between $\texttt{<<no parses (char 2): sm*** >>}$ and any other initial algebra of the functor F is an isomorphism. Suppose $\texttt{<<no parses (char 2): tm*** >>}$: $F(b) \to b$ is another initial algebra of the functor F. By initiality of these two algebras we know there exists a unique algebra homomorphism $\texttt{<<no parses (char 2): fm*** >>} : a \to b$ between $\texttt{<<no parses (char 2): sm*** >>}$ and $\texttt{<<no parses (char 2): tm*** >>}$. Similarly we know there exists a unique algebra homomorphism $\texttt{<<no parses (char 2): fm***' >>} : b \to a$ between $\texttt{<<no parses (char 2): sm*** >>}$ and $\texttt{<<no parses (char 2): tm*** >>}$. Hence we have the following diagrams:



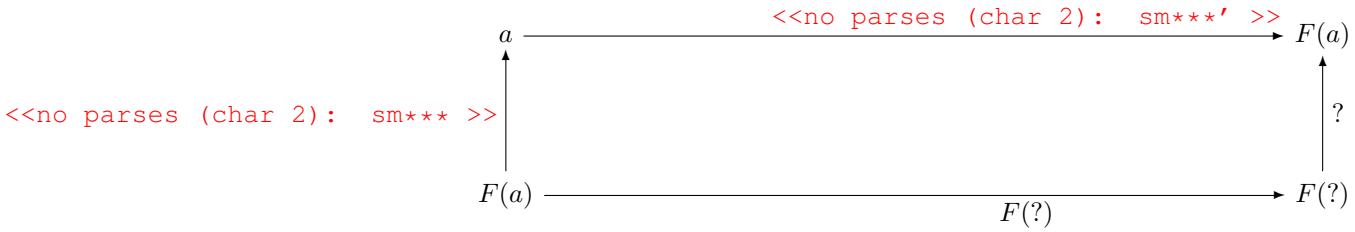
and



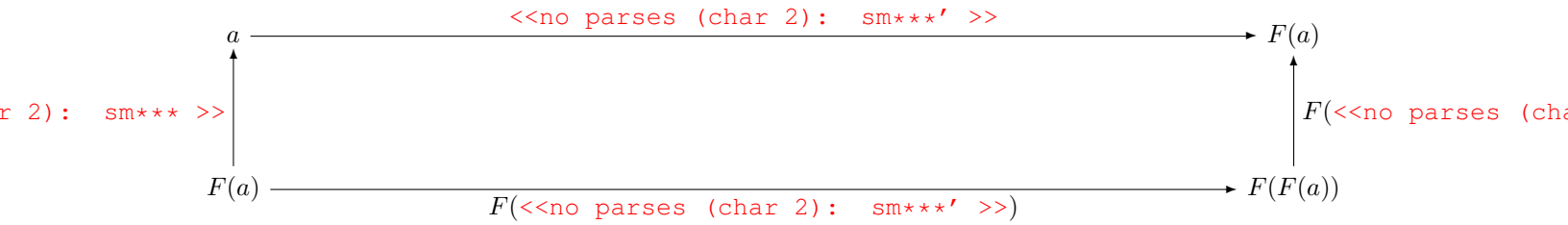
Now we know by initiality that <<no parses (char 2): fm⋆⋆⋆ >> and <<no parses (char 2): fm⋆⋆⋆′ >> are unique and hence their compositions <<no parses (char 2): fm⋆⋆⋆ >>∘<<no parses (char 2): fm⋆⋆⋆′ >> : $a \to a$ and <<no parses (char 2): fm⋆⋆⋆′ >> ∘ <<no parses (char 2): fm⋆⋆⋆ >> : $b \to b$ are also unique. Thus, it must be the case that <<no parses (char 2): fm⋆⋆⋆ >>∘<<no parses (char 2): fm⋆⋆⋆′ id_a and <<no parses (char 2): fm⋆⋆⋆′ >> ∘ <<no parses (char 2): fm⋆⋆⋆ >> $= \mathsf{id}_b$. So <<no parses (char 2): fm⋆⋆⋆ >> and <<no parses (char 2): fm⋆⋆⋆′ >> are mutual inverses. Therefore, <<no parses (char 2): fm⋆⋆⋆ >> is an isomorphism. □

Lemma 13 (Initial Algebras are Isomorphisms)**.** *If* <<no parses (char 2): sm⋆⋆⋆ >> : $F(a) \to a$ *is an initial algebra for some functor* F*, then it has an inverse* <<no parses (char 2): sm⋆⋆⋆ >>$^{-1}$: $a \to F(a)$.

Proof. Suppose <<no parses (char 2): sm⋆⋆⋆ >> : $F(a) \to a$ is an initial algebra for the functor F. We must show there exists an inverse <<no parses (char 2): sm⋆⋆⋆ >>$^{-1}$: $a \to F(a)$ of <<no parses (char 2): sm This can be shown by cleverly finding another algebra such that <<no parses (char 2): sm⋆⋆⋆ >>$^{-1}$ is the algebra homomorphism between <<no parses (char 2): sm⋆⋆⋆ >> and this other algebra. To do this we must fill in the holes of the following diagram:



So our goal is to find an algebra <<no parses (char 2): tm⋆⋆⋆ >> : $F(?) \to F(a)$ such that there is an algebra homomorphism <<no parses (char 2): sm⋆⋆⋆′ >> : $a \to F(a)$, and the above diagram commutes. Choose $F($<<no parses (char 2): sm⋆⋆⋆ >>$) : F(F(a)) \to F(a)$. Hence, we obtain the following diagram:

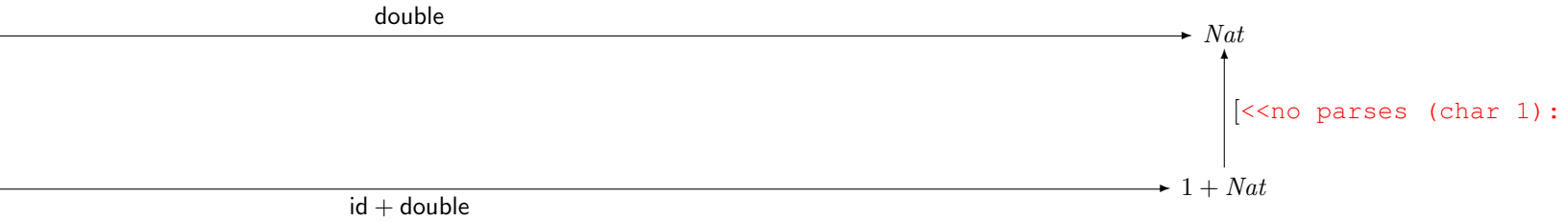


Now we know <<no parses (char 2): sm⋆⋆⋆ >> : $F(a) \to a$ is an initial algebra, and that $F($<<no parses (char 2) $F(F(a)) \to F(a)$ is clearly an algebra. Thus, by initiality (Definition 10) we know that <<no parses (char 2): sm⋆⋆⋆′ >> : $a \to F(a)$ must exist and is an algebra homomorphism. Therefore, it is unique by Definition 9. Hence, <<no parses (char 2): <<no parses (char 2): sm⋆⋆⋆′ >> : $F(a) \to F(a)$ is unique. Furthermore, <<no parses (char 2): sm⋆⋆⋆′ > <<no parses (char 2): sm⋆⋆⋆ >> $= \mathsf{id}_{F(a)}$ and <<no parses (char 2): sm⋆⋆⋆ >>∘<<no parses (char id_a. Therefore, we obtain the following:

$$\texttt{<<no parses (char 2): sm***' >>} \circ \texttt{<<no parses (char 2): sm*** >>} = \mathsf{id}_{F(a)} = F(\mathsf{id}_a) = F(\texttt{<<no parses (char} = F(\texttt{<<no parses (char}$$

Therefore, the inverse of <<no parses (char 2): sm*** >> is <<no parses (char 2): sm***' >>. □

Since our main example has been the natural number initial algebra $[\texttt{<<no parses (char 1): z***m >>}, \texttt{<<no parses}$ $1 + Nat \to Nat$ we will continue developing this example by showing how to use initiality to define functions on the natural numbers by induction. Consider the doubling function, double, which takes a natural number as input, and if it is zero returns zero, otherwise, it returns the number summed with itself. We can define this function by induction using initiality as follows.



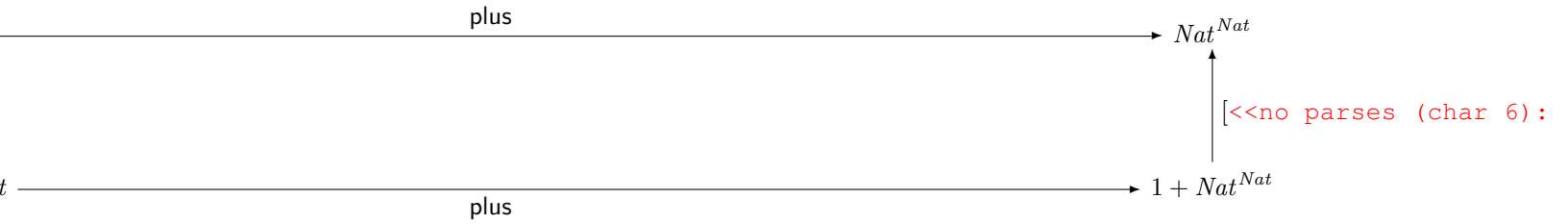
Initiality tells us that double : $Nat \to Nat$ is unique and exists. Furthermore it tells us the following:

$$\mathsf{double} \circ \texttt{<<no parses (char 1): z***m >>} = \texttt{<<no parses (char 1): z***m >>} \circ \mathsf{id} = \texttt{<<no parses}$$
$$\mathsf{double} \circ \texttt{<<no parses (char 2): sm*** >>} = \texttt{<<no parses (char 2): sm*** >>} \circ \texttt{<<no parses (ch}$$

This is equivalent to the following:

$$\mathsf{double}(\texttt{<<no parses (char 1): z***m >>}(*)) = \texttt{<<no parses (char 1): z***m >>}(*)$$
$$\mathsf{double}(\texttt{<<no parses (char 2): sm*** >>}(n)) = \texttt{<<no parses (char 2): sm*** >>}(\texttt{<<no parses (c}$$

Therefore inductively defined functions are no more than algebra homomorphisms between initial algebras!



Initiality tells us that plus : $Nat \to Nat^{Nat}$ is unique and exists. Furthermore it tells us the following:

$$\mathsf{plus} \circ \texttt{<<no parses (char 1): z***m >>} = \texttt{<<no parses (char 2): \f***m.h(\xm.xm) >>} \circ \mathsf{plus}$$
$$\mathsf{plus} \circ \texttt{<<no parses (char 2): sm*** >>} = \texttt{<<no parses (char 2): \f***m.h(\xm.h(sm(fm(xm)))}$$

This is equivalent to the following:

$$\mathsf{plus}(\texttt{<<no parses (char 1): z***m >>}(*)) = (\texttt{<<no parses (char 2): \f***m.h(\xm.xm) >>})(\mathsf{plus}(<$$
$$\mathsf{plus}(\texttt{<<no parses (char 2): sm*** >>}(n)) = (\texttt{<<no parses (char 2): \f***m.h(\xm.h(sm(fm(xm)}$$

1.6 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 $\mathsf{head} : A \to \mathbb{N}$ and $\mathsf{next} : A \to A$ are the morphisms, where A is some fixed set. Now notice that for any $a \in A$ 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 $a \in A$. 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 A.

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

Definition 14 (Coalgebra)**.** *A **coalgebra** or F**-coalgebra** is a pair $(F, \texttt{<<no parses (char 2): sm*** >>}_A)$ with respect to some category C consisting of an object A called the carrier object of the coalgebra, an endofunctor $F : C \to C$, and a morphism $\texttt{<<no parses (char 2): sm*** >>}_A : A \to F(A)$ called the structure of the coalgebra.*

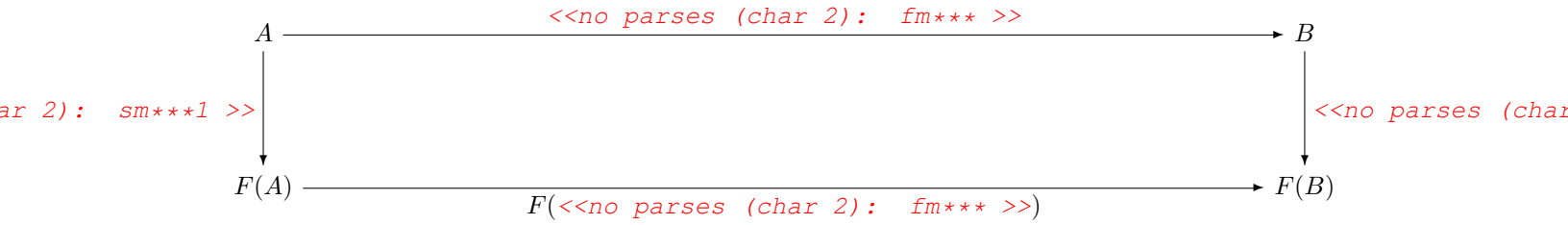
It is often the case that coalgebras are denoted simply by specifying the type of the structure of the coalgebra. That is $\texttt{<<no parses (char 2): sm*** >>}_A : A \to F(A)$. This is precise, because it gives all of the elements of a coalgebra, but the category, but this is usually determinable from the context. So, in our example above we can see that $\langle \mathsf{head}, \mathsf{next} \rangle : A \to F(A)$ 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 $\langle \hat{\mathsf{head}}, \hat{\mathsf{next}} \rangle : B \to F(B)$ with respect to the category Set. Suppose further that there is a function $\texttt{<<no parses (char 2): fm*** >>} : A \to B$ such that $\hat{\mathsf{head}} \circ \texttt{<<no parses (char 2): fm*** >>} = \mathsf{head}$ and $\hat{\mathsf{next}} \circ \texttt{<<no parses (char 2): fm*** >>} = \texttt{<<no parses (char 2): fm*** >>} \circ \mathsf{next}$. In this case $\texttt{<<no parses (char 2): fm*** >>}$ is called a homomorphism between coalgebras.

Definition 15 (Homomorphism)**.** *A **homomorphism** between two coalgebras $\texttt{<<no parses (char 2): sm***1 >>} : A \to F(A)$ and $\texttt{<<no parses (char 2): sm***2 >>} : B \to F(B)$ is a morphism $\texttt{<<no parses (char 2): fm***} : A \to B$ such that the following diagram commutes:*



The above diagram expresses that $\texttt{<<no parses (char 2): fm*** >>}$ commutes with the operations of the two coalgebras.

Definition 16 (Final Coalgebras)**.** *A **final coalgebra** $\texttt{<<no parses (char 2): sm***1 >>} : A \to F(A)$ is one such that for any other coalgebra $\texttt{<<no parses (char 2): sm***2 >>} : B \to F(B)$ 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