ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

A Categorical Characterization of Untyped Lambda Calculus

Master's Thesis

Daniel Schmitter

March 1, 2013

Advisor: Prof. Dr. Damien Calaque
Department of Mathematics, ETH Zurich

Abstract The purpose of this thesis is to describe untyped lambda calculus as an endofunctor on small sets and to prove that it is an initial object in the category of exponential monads.

Acknowledgements I thank my advisor, Prof. Dr. Damien Calaque, for everything he did for me during the last months, such as suggesting this topic, offering a lot of his time, his insight, the pleasant conversations we had during the time of writing this thesis, his ideas that helped me seeing things from different angles, and a lot more. Further I would like to thank the advisor of my previous theses, Prof. Dr. Clemens Fuchs, for his support during my studies and everyone else who helped me getting to this point.

Keywords category theory, characterization, endofunctor, exponential monad, initial object, lambda calculus, lambda-term, simply typed, operad, untyped

Contents

Chapter 1

Introduction

The main goal of this thesis is to prove untyped lambda calculus initial in a specific category. To understand the statements and proofs herein the reader should have basic knowledge of lambda calculus (as it can be found in [BB00] for example) and category theory (especially functors, natural transformations, monads, slice categories, and cartesian closed categories).

This thesis is based on [Zsi06], which is itself based on [HM05]. The statements therein are the same as ours. However we prove the main theorem without the help of the theorem prover "Coq", which was used in the above papers.

We try to motivate our definitions and proofs in such a way that the reader can not just understand what we did but also why. The thesis is therefore a bit longer than it would need to be but we feel that it is better for the understanding of the subject to know why we do things the way we do them. This also means that we sometimes show that the obvious way has to fail. We always mention when inserting such a part as not to confuse the reader.

On the way to our goal we proceed step by step. Chapter 2 gives all the definitions needed for the category we are going to work with, which is the category of exponential monads. We don't mention properties that we are not going to use. For further facts about monads and modules such as pull-backs we refer the reader to [Zsi06].

In the following chapter we describe untyped lambda calculus in these terms. Afterwards we show in Chapter 4 that all objects in the category of exponential monads (or under even weaker conditions) satisfy some properties that are known for lambda calculus. Finally we prove the main theorem in Chapter 5.

The last chapter then gives a small summery of our work and shows how one can proceed with this result at hand. We explain for example how one could show a similar statement for simply typed lambda calculus. Additionally we give some ideas for an approach by operads instead of monads.

Notation

- By $\mathbb{N}$ we denote the natural numbers according to the Dedekind-Peano axioms, i.e. we have $0 \in \mathbb{N}$.
- By **Set** we denote the category of all small sets (as defined in [ML98]). By a formulation like "$X \in \mathbf{Set}$" we mean an object in that category. If we talk about morphisms we state so explicitly.

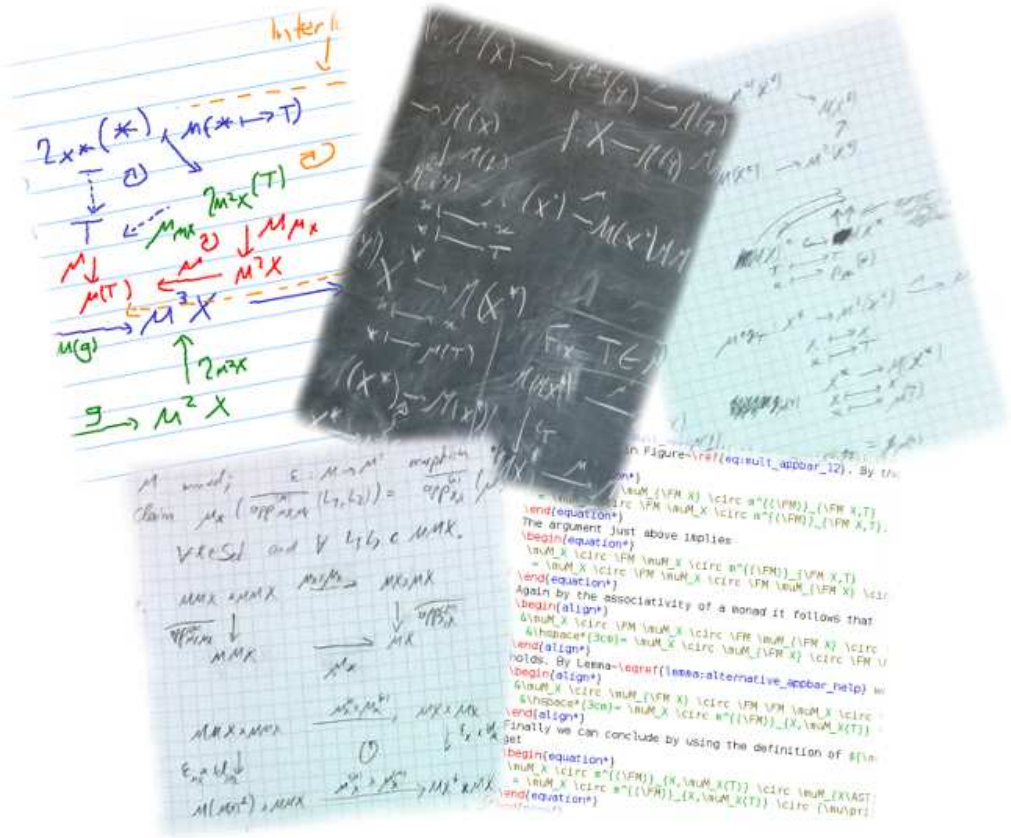
- As we are not going to consider large sets we just write "set" instead of "small set".
- For $X, Y \in \mathbf{Set}$ we define their disjoint union by

$$X \amalg Y \coloneqq (X \times \{0\}) \cup (Y \times \{1\}).$$

  We use the following (abuse) of notation: For all $x \in X$ we denote its image in $X \amalg Y$ by x and not by $j_X(x)$, where $j_X : X \to X \amalg Y$ is the canonical inclusion of X into $X \amalg Y$, and similarly for all $y \in Y$.
- If no confusion arises we omit parentheses for functors, i.e. given two categories $\mathscr{C}$ and $\mathscr{D}$ and a functor $\mathrm{F} : \mathscr{C} \to \mathscr{D}$ we write $\mathrm{F}\,C$ for the image under F of any object $C \in \mathscr{C}$ and $\mathrm{F}\,g$ for the image under F of any morphism $g \in \mathscr{C}$. However for a map $f : X \to Y$ between two sets X and Y we write $f(x)$ for the image under f of any $x \in X$.
- When we are working with λ-terms without any equivalence relation involved we write $L = K$ to state that the two λ-terms are equal. As soon as an equivalence relation is involved we use the notation of [Sel07], i.e. $L = K$ denotes that L and K are two representatives of the same equivalence class and $L \equiv K$ denotes syntactic equivalence (they are the same "symbol by symbol").
- For λ-terms we use the same rules for omitting parentheses as [Bar84], i.e. dropping outermost parentheses, taking application as left associative, and abstraction as right associative and binding as far to the right as possible.
- When talking about α-, β-, and/or η-conversions we will talk about the equivalence relations they generate, however this is a (usual) misnomer: As described in [Bar84, p. 51] these equivalence relations are built step by step where one step involves a compatible closure, i.e. the relations are actually congruence relations (with respect to application and abstraction). For details see [Bar84, Lemma 3.1.6].

Work in Progress (Just for Fun)

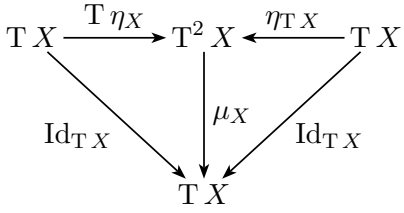
The picture below shows the evolution of a proof in the case of Lemma 4.5. This includes digital sketches on the go and sketches on the blackboard and on paper (top row). These were followed by a (more or less) readable sketch on paper and finally everything was typeset such that you get what you see right now.



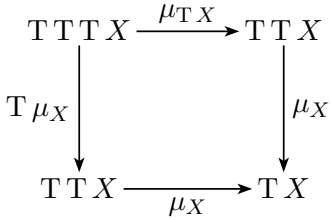
Categorical Preliminaries

This chapter introduces the necessary definitions for the category we are going to work with in this thesis. Most of the definitions should be familiar but some might not. This chapter mainly serves to establish the notation we use.

Let $\mathscr{C}$ be any category, a *monad over* $\mathscr{C}$ is a triple (T, η, μ) where $\mathrm{T}\colon \mathscr{C} \to \mathscr{C}$ is an endofunctor and $\eta\colon \mathrm{Id} \dot{\to} \mathrm{T}$ and $\mu\colon \mathrm{T}^2 \dot{\to} \mathrm{T}$ are natural transformations such that the two diagrams



(2.1)



(2.2)

are commutative for all objects $X \in \mathscr{C}$. Here $\mathrm{Id}\colon \mathscr{C} \to \mathscr{C}$ denotes the identity functor on $\mathscr{C}$. The natural transformations η and μ are called the *unit* and the *multiplication* of the monad. Condition (2.2) is referred to as the *associativity* of the monad (or the associativity of μ). If η and μ are understood we call T itself a monad. All monads in this thesis (except Chapter 6) are over **Set**, we are often not mentioning this explicitly.

For all $X \in \mathbf{Set}$ we denote by X^* the disjoint union of X with a one-element set $*$. Unless stated otherwise we use "$*$" to denote the one-element set as well as its unique element. The canonical injection of X into X^* is denoted by ι_X. For a map of sets $f\colon X \to Y$ we denote by $f^*\colon X^* \to Y^*$ the map satisfying $f^*|_X = \iota_Y \circ f$ and $f(*) = *$.

Given any monad (T, η, μ) over **Set**, we define its *derived monad* T' for all $X \in \mathbf{Set}$ by

$$\mathrm{T}'\,X \coloneqq \mathrm{T}\,X^* (= \mathrm{T}(X^*))$$

and on all morphisms of sets $f\colon X \to Y$ by

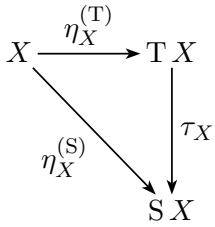
$$\mathrm{T}' f \coloneqq \mathrm{T}\, f^*(= \mathrm{T}(f^*)).$$

The unit and the multiplication are given for all $X \in \mathbf{Set}$ by

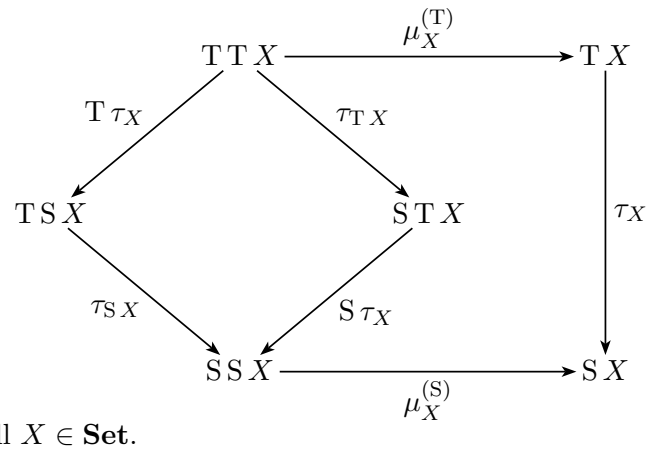
$$\begin{aligned} \eta'_X &\coloneqq \mathrm{T}\, \iota_X \circ \eta_X \\ \mu'_X &\coloneqq \mu_{X^*} \circ \mathrm{T}\, d_X, \end{aligned}$$

where $d_X\colon (\mathrm{T}' X)^* \to \mathrm{T}' X$ is given by $d_X|_{\mathrm{T}' X} = \mathrm{Id}_{\mathrm{T}' X}$ and $d_X(*) = \eta_{X^*}(*)$. By [Zsi06, Remark 2.5] the derived monad of a monad is again a monad.

Let $(\mathrm{T}, \eta^{(\mathrm{T})}, \mu^{(\mathrm{T})})$ and $(\mathrm{S}, \eta^{(\mathrm{S})}, \mu^{(\mathrm{S})})$ be two monads over $\mathbf{Set}$. A *morphism of monads* is a natural transformation $\tau\colon \mathrm{T} \dot{\to} \mathrm{S}$ such that the two diagrams



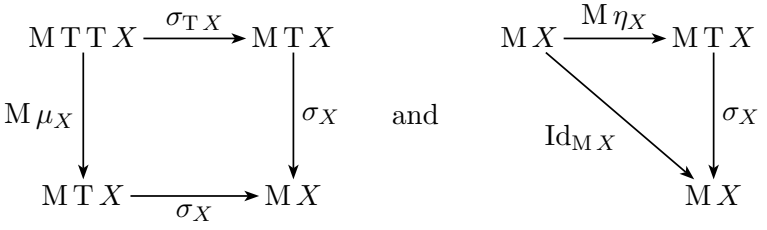
(2.3)



(2.4)

commute for all $X \in \mathbf{Set}$.

A *right module* over a monad (T, η, μ) over $\mathbf{Set}$ is a pair (M, σ), where M is an endofunctor on $\mathbf{Set}$ and $\sigma\colon \mathrm{M}\,\mathrm{T} \dot{\to} \mathrm{M}$ is a natural transformation such that



(2.5)

commute for all $X \in \mathbf{Set}$. Sometimes, e.g. in [HM04] and [Zsi06], it is only required that the diagram on the left commutes. However also asking for the right diagram to commute justifies the terminology "module" (as it is known in

ring theory) and does not introduce any restrictions to the statements in this thesis. We are not going to consider left modules, therefore a right module will just be referred to as a *module*. As with monads we often call M a module without mentioning σ explicitly. By setting $\sigma := \mu$ we see by (2.1) and (2.2) that T is a module over itself, called the *tautological module*.

Given a monad (T, η, μ) over **Set** and a T-module (M, σ), we construct its *derived module* (M', σ') as follows: M' is given for all $X \in \mathbf{Set}$ by

$$\mathrm{M}'\,X := \mathrm{M}\,X^*,$$

for all morphisms $f\colon X \to Y$ in **Set** by

$$\mathrm{M}'\,f := \mathrm{M}\,f^*,$$

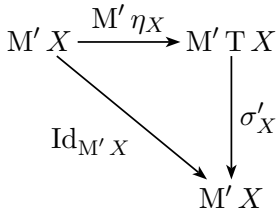
and σ' is given for all $X \in \mathbf{Set}$ by

$$\sigma'_X := \sigma_{X^*} \circ \mathrm{M}\,g_X,$$

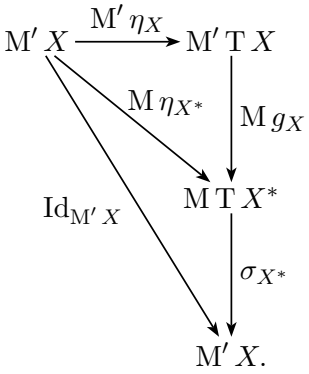
where $g_X\colon (\mathrm{T}\,X)^* \to \mathrm{T}\,X^*$ is defined by $g_X|_{\mathrm{T}\,X} = \mathrm{T}\,\iota_X$ and $g_X(*) = \eta_{X^*}(*)$.

Lemma 2.1 *Let* (T, η, μ) *be any monad over* **Set** *and let* (M, σ) *be any* T*-module. Then the derived module* (M', σ') *is itself a* T*-module.*

Proof We need to prove the diagrams corresponding to (2.5) commutative for all $X \in \mathbf{Set}$. The commutativity of the left diagram is shown in [Zsi06, Remark 2.11]. For the right diagram let $X \in \mathbf{Set}$ be arbitrary. We need to prove



commutative. Using the definitions this diagram becomes



The lower triangle commutes as (M, σ) is a T-module, hence it suffices to prove the upper triangle commutative. By functoriality it is even enough to prove

$$\begin{array}{ccc} X^* & \xrightarrow{\eta^*_X} & (\mathrm{T}\,X)^* \\ & \searrow^{\eta_{X^*}} & \downarrow{\scriptstyle g_X} \\ & & \mathrm{T}\,X^* \end{array}$$

commutative. Using the definitions of η^*_X and g_X we easily see the commutativity for $* \in X^*$. The commutativity for all $x \in X$ follows in the same way using naturality of η, i.e. that

$$\begin{array}{ccc} X & \xrightarrow{\eta_X} & \mathrm{T}\,X \\ {\scriptstyle \iota_X}\downarrow & & \downarrow{\scriptstyle \mathrm{T}\,\iota_X} \\ X^* & \xrightarrow[\eta_{X^*}]{} & \mathrm{T}\,X^* \end{array}$$

commutes, and that the image of ι_X is (contained in) $X \hookrightarrow X^*$. □

Given two T-modules $(\mathrm{M}, \sigma^{(\mathrm{M})})$ and $(\mathrm{N}, \sigma^{(\mathrm{N})})$, a *morphism of (right)* T*-modules* is a natural transformation $\varphi\colon \mathrm{M} \dot{\to} \mathrm{N}$ such that

$$\begin{array}{ccc} \mathrm{M}\,\mathrm{T}\,X & \xrightarrow{\sigma^{(\mathrm{M})}_X} & \mathrm{M}\,X \\ {\scriptstyle \varphi_{\mathrm{T}\,X}}\downarrow & & \downarrow{\scriptstyle \varphi_X} \\ \mathrm{N}\,\mathrm{T}\,X & \xrightarrow[\sigma^{(\mathrm{N})}_X]{} & \mathrm{N}\,X \end{array} \tag{2.6}$$

commutes for all $X \in \mathbf{Set}$. We get that the canonical injection $\mathrm{M}\,\iota\colon \mathrm{M} \hookrightarrow \mathrm{M}'$ is a morphism of modules (cf. [Zsi06, Lemma 2.12]).

Given two T-modules $(\mathrm{M}, \sigma^{(\mathrm{M})})$ and $(\mathrm{N}, \sigma^{(\mathrm{N})})$, the product of M and N in the category of all functors from **Set** to **Set** is again a T-module by setting

$$\sigma^{(\mathrm{M}\times\mathrm{N})}_X \coloneqq \sigma^{(\mathrm{M})}_X \times \sigma^{(\mathrm{N})}_X$$

for all $X \in \mathbf{Set}$ (cf. [Zsi06, Remark 2.13]).

An *exponential monad* $(\mathrm{T}, \varepsilon)$ over **Set** is a pair consisting of a monad (T, η, μ) together with an isomorphism ε from the tautological T-module to its derived module T'.

Given two exponential monads $(\mathrm{T}, \varepsilon^{(\mathrm{T})})$ and $(\mathrm{S}, \varepsilon^{(\mathrm{S})})$, a *morphism of exponential monads* is a natural transformation $\nu\colon \mathrm{T} \dot{\to} \mathrm{S}$ such that both squares (the

one with $\varepsilon_X^{(\mathrm{T})}$ and $\varepsilon_X^{(\mathrm{S})}$ and the one with ${\varepsilon_X^{(\mathrm{T})}}^{-1}$ and ${\varepsilon_X^{(\mathrm{S})}}^{-1}$) in

$$\begin{array}{ccc} \mathrm{T}\,X & \xrightarrow{\nu_X} & \mathrm{S}\,X \\ {\scriptstyle \varepsilon_X^{(\mathrm{T})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{T})}}^{-1}} & & {\scriptstyle \varepsilon_X^{(\mathrm{S})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{S})}}^{-1}} \\ \mathrm{T}'\,X & \xrightarrow[\nu_{X^*}]{} & \mathrm{S}'\,X \end{array}$$

commute for all $X \in \mathbf{Set}$.

Lemma 2.2 *Exponential monads and morphisms of exponential monads form the objects and morphisms of a category.*

Proof Composition of morphisms is just vertical composition of natural transformations (cf. [ML98, p. 40]), therefore we immediately have the existence of an identity and the associativity of composition. The only thing left to show is that everything commutes with the isomorphisms of exponential monads. Therefore let $(\mathrm{P}, \varepsilon^{(\mathrm{P})})$, $(\mathrm{Q}, \varepsilon^{(\mathrm{Q})})$, $(\mathrm{R}, \varepsilon^{(\mathrm{R})})$, and $(\mathrm{S}, \varepsilon^{(\mathrm{S})})$ be any exponential monads. For the identity this means that for all morphisms of exponential monads $\nu\colon \mathrm{P} \dot{\to} \mathrm{Q}$ and $\tau\colon \mathrm{Q} \dot{\to} \mathrm{R}$ the four composed rectangles in

$$\begin{array}{ccccccc} \mathrm{P}\,X & \xrightarrow{\nu_X} & \mathrm{Q}\,X & \xrightarrow{\mathrm{Id}_{\mathrm{Q}\,X}} & \mathrm{Q}\,X & \xrightarrow{\tau_X} & \mathrm{R}\,X \\ {\scriptstyle \varepsilon_X^{(\mathrm{P})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{P})}}^{-1}} & & {\scriptstyle \varepsilon_X^{(\mathrm{Q})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{Q})}}^{-1}} & & {\scriptstyle \varepsilon_X^{(\mathrm{Q})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{Q})}}^{-1}} & & {\scriptstyle \varepsilon_X^{(\mathrm{R})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{R})}}^{-1}} \\ \mathrm{P}'\,X & \xrightarrow[\nu_{X^*}]{} & \mathrm{Q}'\,X & \xrightarrow[\mathrm{Id}_{\mathrm{Q}\,X^*}]{} & \mathrm{Q}'\,X & \xrightarrow[\tau_{X^*}]{} & \mathrm{R}'\,X \end{array}$$

are commutative for all $X \in \mathbf{Set}$. This is obvious as by the definition of morphisms of exponential monads all three small squares are commutative for all $X \in \mathbf{Set}$. For associativity let $\nu : \mathrm{P} \dot{\to} \mathrm{Q}$, $\tau : \mathrm{Q} \dot{\to} \mathrm{R}$, and $\omega : \mathrm{R} \dot{\to} \mathrm{S}$ be morphisms of exponential monads. One can then take a look at

$$\begin{array}{ccccccc} \mathrm{P}\,X & \xrightarrow{\nu_X} & \mathrm{Q}\,X & \xrightarrow{\tau_X} & \mathrm{R}\,X & \xrightarrow{\omega_X} & \mathrm{S}\,X \\ {\scriptstyle \varepsilon_X^{(\mathrm{P})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{P})}}^{-1}} & & {\scriptstyle \varepsilon_X^{(\mathrm{Q})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{Q})}}^{-1}} & & {\scriptstyle \varepsilon_X^{(\mathrm{R})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{R})}}^{-1}} & & {\scriptstyle \varepsilon_X^{(\mathrm{S})}}\downarrow \uparrow{\scriptstyle {\varepsilon_X^{(\mathrm{S})}}^{-1}} \\ \mathrm{P}'\,X & \xrightarrow[\nu_{X^*}]{} & \mathrm{Q}'\,X & \xrightarrow[\tau_{X^*}]{} & \mathrm{R}'\,X & \xrightarrow[\omega_{X^*}]{} & \mathrm{S}'\,X, \end{array}$$

where "everything" commutes for all $X \in \mathbf{Set}$ as this is the case for all small squares. □

We are now prepared to move on to give a description of untyped lambda calculus in these terms.

Chapter 3

Untyped Lambda Calculus as an Exponential Monad

In this chapter we show how untyped lambda calculus can be seen as an exponential monad. We do so by first defining an endofunctor on **Set** not involving any equivalence relations and then realize lambda calculus as this endofunctor with equivalence relations applied together with a suitable unit and multiplication. The last step is then a justification why this endofunctor describes lambda calculus.

This Chapter contains parts (such as Section 3.1) that describe ideas that are not going to work. We suggest the reader not to skip them as they might still give some intuition of what is going on.

3.1 Direct Approach

We first give a description of a "direct" approach of defining lambda calculus as an exponential monad and show why it fails. We try to define an endofunctor SLC on **Set** which assigns to every set the set of λ-terms with free variables in that set and which assigns to every function between sets the function that renames the variables in the λ-terms according to this function. However this approach has to fail as soon as we introduce equivalence classes: For example consider lambda calculus over two sets X and Y such that $x_1, x_2, x_3 \in X$ and $y_1, y_2, y_3 \in Y$. Given a map $f : X \to Y$ satisfying $f(x_1) = f(x_2) = y_1$ and $f(x_3) = y_3$ we get the following results by renaming:

$$\lambda x_3.x_2 \mapsto \lambda y_3.y_1 \quad \text{and} \quad \lambda x_1.x_2 \mapsto \lambda y_1.y_1.$$

The two left sides are equivalent by an $\alpha-$conversion ($\lambda x_3.x_2 = \lambda x_1.x_2$) however the right sides are clearly not ($\lambda y_3.y_1$ is a constant but $\lambda y_1.y_1$ is the identity). Therefore this map would not be well-defined on equivalence classes. The idea is now to introduce a new set (called D) to which all bound variables get mapped first and then apply f only to the free variables (and not changing the bound ones).

3.2 Action on Objects

Formally we define SLC as follows: For the rest of this thesis let D denote a fixed countably infinite set. Further fix an order on D such that D is isomorphic to

the natural numbers in the category of totally ordered sets (i.e. it has a smallest element and for every element a unique successor such that every element, except the smallest one, is the successor of another element). We define the action of the functor $\mathrm{SLC}\colon \mathbf{Set} \to \mathbf{Set}$ for all $X \in \mathbf{Set}$ by

$$X \mapsto \{L \mid L \text{ is a } \lambda\text{-term with variables in } X \amalg D \text{ with } \mathrm{FV}(L) \subseteq X\},$$

where $\mathrm{FV}(L)$ denotes the set of free variables of L. In the following we use letters from the end of the alphabet (mostly x, y, z) to denote variables that may be free, letters from the beginning of the alphabet (mostly c, d) for variables belonging to D, and intermediate letters (mostly p, q) to denote variables that might come from either set.

3.3 Preparations for Action on Morphisms

Before we can define the action of SLC on morphisms we need some additional notations and statements that make our presentation simpler (although it might not seem so at first).

For all $X \in \mathbf{Set}$ we denote by $\mathrm{var}_X \colon X \to \mathrm{SLC}\, X$ the insertion of variables, i.e. every $x \in X$ gets mapped to the variable corresponding to x. The map that sends two λ-terms in $\mathrm{SLC}\, X$ for some $X \in \mathbf{Set}$ to the application of the two λ-terms is denoted by $\mathrm{app}_X \colon \mathrm{SLC}\, X \times \mathrm{SLC}\, X \to \mathrm{SLC}\, X$.

The straight forward approach to define $\mathrm{SLC}\, f : \mathrm{SLC}\, X \to \mathrm{SLC}\, Y$ for all morphisms $f \colon X \to Y$ in $\mathbf{Set}$ would be to do so recursively for all $L \in \mathrm{SLC}\, X$ by

$$L \mapsto \begin{cases} \mathrm{var}_Y(f(x)), & \text{if } L = \mathrm{var}_X(x) \text{ with } x \in X, \\ \mathrm{app}_Y(\mathrm{SLC}\, f(L_1), \mathrm{SLC}\, f(L_2)), & \text{if } L = \mathrm{app}_X(L_1, L_2) \\ & \text{with } L_1, L_2 \in \mathrm{SLC}\, X, \\ \lambda\, \mathrm{var}_Y(\tilde{f}(q)).(\mathrm{SLC}\, f(L')), & \text{if } L = \lambda q.L' \text{ with } q \in X \amalg D \\ & \text{and } L' \in \mathrm{SLC}\, X, \end{cases}$$

where $\tilde{f} \colon X \amalg D \to Y \amalg D$ is given by $\tilde{f} \coloneqq f \amalg \mathrm{Id}_D$. However this approach has to fail as if we have $q \in D$ in the last case, then q might be free in L' which would imply $L' \notin \mathrm{SLC}(X)$.

To give a well-defined action on morphisms we need some additional constructions. We say that a λ-term is in *canonical form* if all bound variables used to build it belong to D and are in ascending order (from inside out) starting with the smallest element in D. For all $X \in \mathbf{Set}$ and all $L \in \mathrm{SLC}\, X$ there exists a unique λ-term in canonical form in $\mathrm{SLC}\, X$ which is α-equivalent to L. This λ-term is denoted by $(L)_{\mathrm{C}_X}$. For all $X \in \mathbf{Set}$ we denote by $\mathrm{C}_X \colon \mathrm{SLC}\, X \to \mathrm{SLC}\, X$ the map which assigns to each $L \in \mathrm{SLC}\, X$ its canonical form. It is not easy (if at all possible) to give a closed or recursive formula for this process, however one can write an algorithm that computes this function such that we see that it is well-defined. An example of how such an algorithm could work is given in Section 3.9.

We could now define $\mathrm{SLC}\, f \colon \mathrm{SLC}\, X \to \mathrm{SLC}\, Y$ for all $f \colon X \to Y$ in **Set** by

$$L \mapsto \tilde{f}((L)_{\mathrm{C}_X})$$

for all $L \in \mathrm{SLC}\, X$, where $\tilde{f}$ is again given by $\tilde{f} \coloneqq f \amalg \mathrm{Id}_D$. For $c \coloneqq \min\{d \mid d \in D\}$ the example from Section 3.1 becomes

$$\lambda x_3.x_2 \mapsto \lambda c.x_2 \mapsto \lambda c.y_1 \quad \text{and} \quad \lambda x_1.x_2 \mapsto \lambda c.x_2 \mapsto \lambda c.y_1.$$

One could now show that this map is actually constant on δ-equivalence classes for $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$. As we are not going to take this as our definition of $\mathrm{SLC}\, f$ a proof is omitted. The reason for not taking this approach is that it is too artificial in the sense that the definition doesn't involve the structure of λ-terms. For example if a λ-term is built by an application we would like to say that the image of this λ-term under $\mathrm{SLC}\, f$ is determined by the images of these two λ-terms under $\mathrm{SLC}\, f$. The advantage of the (more technical) approach below is that it gives a way of proving statements inductively, which would not be possible by the way described above. In the following we make this precise but the reader should keep in mind that our definition below is (up to α-equivalence) just what we described right now.

We define for all $X \in \mathbf{Set}$ a map $\mathrm{abs}_X \colon \mathrm{SLC}'\, X \to \mathrm{SLC}\, X$ which is given for all $L \in \mathrm{SLC}'\, X$ by

$$\mathrm{abs}_X(L) \coloneqq r_X((\lambda * .L)_{\mathrm{C}_{X^*}}),$$

where $r_X \colon \mathrm{SLC}'\, X \to \mathrm{SLC}\, X$ is given for all $K \in \mathrm{SLC}'\, X$ by

$$K \mapsto \begin{cases} K, & \text{if } \mathrm{FV}(K) \cup \mathrm{BV}(K) \subset X \amalg D, \\ \lambda c.c, & \text{otherwise,} \end{cases}$$

where $\mathrm{BV}(K)$ denotes the bound variables in K, $c \coloneqq \min\{d \mid d \in D\}$, and the K on the right side stands for the λ-term in $\mathrm{SLC}\, X$ that is built in the same way as K is in $\mathrm{SLC}'\, X$, i.e. has the same form and the same variables (with respect to the inclusion of variables in the corresponding disjoint unions). Note that for all $L \in \mathrm{SLC}'\, X$ it holds that $\mathrm{FV}((\lambda * .L)_{\mathrm{C}_{X^*}}) \cup \mathrm{BV}((\lambda * .L)_{\mathrm{C}_{X^*}}) \subset X \amalg D$.

In the following lemma we show that for any λ-term $L \in \mathrm{SLC}\, X$ of the form $L = \lambda q.L'$ it holds that $(L)_{\mathrm{C}_X}$ is in the image of abs_X. For simplicity we assume in the following that the corresponding λ-term in $\mathrm{SLC}'\, X$ is in canonical form (which exists by the lemma).

Lemma 3.1 *For all $X \in \mathbf{Set}$ and all $L \in \mathrm{SLC}\, X$ of the form $L = \lambda q.L'$ it holds that $(L)_{\mathrm{C}_X}$ is in the image of abs_X, i.e. there exists a $K \in \mathrm{SLC}'\, X$ such that $\mathrm{abs}_X(K) = (L)_{\mathrm{C}_X}$. Further this K is unique if we force it to be in canonical form.*

Proof Let $X \in \mathbf{Set}$ be arbitrary. We get the needed $K \in \mathrm{SLC}' X$ by chasing $L \in \mathrm{SLC}\, X$ through the following chain of maps:

$$\mathrm{SLC}\, X \xrightarrow{\mathrm{C}_X} \mathrm{SLC}\, X \xrightarrow{s_X} \mathrm{SLC}' X \xrightarrow{\mathrm{C}_{*,X^*}} \mathrm{SLC}' X \xrightarrow{\mathrm{f}_X} \mathrm{SLC}' X.$$

The maps used are the following:

- C_X is the function that maps any λ-term to its canonical form.
- $s_X : \mathrm{SLC}\, X \to \mathrm{SLC}' X$ is the map similar to r_X defined earlier, i.e. it sends any λ-term $L \in \mathrm{SLC}\, X$ to the "same" λ-term in $\mathrm{SLC}' X$ (with respect to the corresponding inclusions into the disjoint unions).
- C_{*,X^*} is C_{X^*} composed with $\mathrm{c}_{*,X} : \mathrm{SLC}' X \to \mathrm{SLC}' X$, which is given for all $L \in \mathrm{SLC}' X$ by

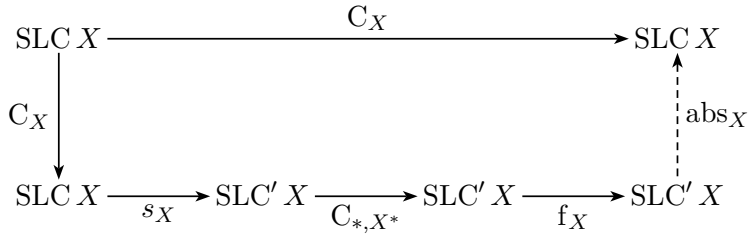
$$L \mapsto \begin{cases} \mathrm{var}_{X^*}(y), & \text{if } L = \mathrm{var}_{X^*}(y) \text{ with } y \in X^*, \\ \mathrm{app}_{X^*}(L_1, L_2), & \text{if } L = \mathrm{app}_{X^*}(L_1, L_2) \text{ with } L_1, L_2 \in \mathrm{SLC}' X, \\ \mathrm{A}_*(L), & \text{if } L = \lambda q.L' \text{ with } q \in D \text{ and } * \notin L', \\ \lambda c.c, & \text{otherwise,} \end{cases}$$

  where $c \coloneqq \min\{d \mid d \in D\}$ and $\mathrm{A}_*(L)$ denotes the λ-term L after the α-conversion that replaces q with $*$ (this is possible due to the assumption that $* \notin L'$). Precomposition with s_X and C_{X^*} guarantees that we land in one of the first three cases.
- $\mathrm{f}_X : \mathrm{SLC}' X \to \mathrm{SLC}' X$ is given for all $L \in \mathrm{SLC}' X$ by

$$L \mapsto \begin{cases} L', & \text{if } L = \lambda * .L' \text{ with } L' \in \mathrm{SLC}' X, \\ \mathrm{var}_{X^*}(\min\{d \mid d \in D\}), & \text{otherwise.} \end{cases}$$

  Precomposition with the previous maps again guarantees that we land in the first case.

The statement that $\mathrm{abs}(K) = (L)_{\mathrm{C}_X}$ holds for $K = \mathrm{f}_X(\mathrm{C}_{*,X^*}(s_X(\mathrm{C}_X(L))))$ is just that abs_X turns



commutative. This is obvious by the definitions of the maps involved.

The last part is easy to see: Assume that there are $K, K' \in \mathrm{SLC}' X$ such that $\mathrm{abs}_X(K) = \mathrm{abs}_X(K')$. As the last two maps in the definition of abs_X are not changing the structure of the λ-terms we get that $\lambda * .K$ is α-equivalent to $\lambda * .K'$. However this implies that K and K' are α-equivalent. The claim then follows as α-equivalent λ-terms have the same canonical form. $\square$

3.4 Main Proof Technique

Most of our proofs (and definitions) will be inductively on the complexity of the λ-terms involved. More precisely we define the *degree* of any untyped λ-term $L \in \mathrm{SLC}\, X$ for all $X \in \mathbf{Set}$ by the function $\deg\colon \mathrm{SLC}\, X \to \mathbb{N}$, which is given by precomposition with C_X of $\widetilde{\deg}\colon \mathrm{SLC}\, X \to \mathbb{N}$ defined for all $K \in \mathrm{SLC}\, X$ by

$$\widetilde{\deg}(K) := \begin{cases} 0, & \text{if } (K)_{\mathrm{C}_X} = K = \mathrm{var}_X(x) \text{ with } x \in X, \\ \widetilde{\deg}(K_1) + \widetilde{\deg}(K_2) + 1, & \text{if } (K)_{\mathrm{C}_X} = K = \mathrm{app}_X(L_1, L_2) \\ & \quad \text{with } K_1, K_2 \in \mathrm{SLC}\, X, \\ \widetilde{\deg}(K') + 1, & \text{if } (K)_{\mathrm{C}_X} = K = \mathrm{abs}_X(K') \\ & \quad \text{with } K' \in \mathrm{SLC}'\, X, \\ 0, & \text{otherwise.} \end{cases}$$

For all $X \in \mathbf{Set}$ and all $k \in \mathbb{N}$ we denote by $\mathrm{SLC}^{\langle k \rangle}\, X$ the set of all $L \in \mathrm{SLC}\, X$ with $\deg(L) = k$.

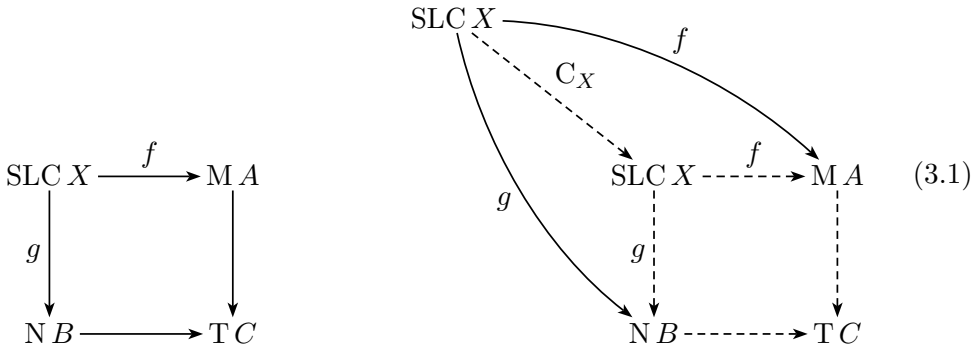
Our proofs usually have the following form: We first show that a statement holds for $\mathrm{SLC}^{\langle 0 \rangle}\, X$ for all $X \in \mathbf{Set}$ (and/or for all morphisms between these sets induced by $f : X \to Y$ in $\mathbf{Set}$). We then assume that the statement holds for $\mathrm{SLC}^{\langle i \rangle}\, X$ for all $X \in \mathbf{Set}$ (and/or for all morphisms between these sets induced by $f : X \to Y$ in $\mathbf{Set}$) for all $i < k$ for some $k \in \mathbb{N} \setminus \{0\}$. If we succeed to prove that given these assumptions the statement also holds for $\mathrm{SLC}^{\langle k \rangle}\, X$ for all $X \in \mathbf{Set}$ (and/or for all morphisms between these sets induced by $f\colon X \to Y$ in $\mathbf{Set}$) we are done as any $L \in \mathrm{SLC}\, X$ belongs to exactly one $\mathrm{SLC}^{\langle j \rangle}\, X$ for some $j \in \mathbb{N}$. As any λ-term is built - up to α-equivalence (see paragraph below) - from variables and the operations app and abs (and any λ-term generated by these operations has a strictly higher degree than the λ-term(s) used to build it) this means that we need to prove the statement for variables (base case) and then for λ-terms built by app or abs as a last step, assuming the statement holds for the λ-terms used therein (induction). As not to make proofs longer than needed we are not going to repeat this argument but just apply this technique.

The formulation of the induction technique above will not really give a proof as abs always yields a term in canonical form but there are other terms that are abstractions in the usual sense of lambda calculus. However this can be fixed in the following way:

Convention For all $X \in \mathbf{Set}$ we *always* assume that any map from $\mathrm{SLC}\, X$ is precomposed with C_X and any map to $\mathrm{SLC}\, X$ is postcomposed with C_X (even if we don't mention it explicitly).

Hence assume we need to prove the diagram on the left in (3.1) (see next page) commutative for some monads M, N, and T, some $A, B, C, X \in \mathbf{Set}$, and some maps f and g. Assuming that the statement holds for terms in canonical form we then know that the dashed part in the right diagram in (3.1) commutes. However as we assume that f and g are some maps precomposed with C_X we see (using

that C_X is idempotent, i.e. that $\mathrm{C}_X \circ \mathrm{C}_X = \mathrm{C}_X$ holds) that the two triangles also commute. Therefore the statement has to hold for all λ-terms in $\mathrm{SLC}\, X$ if we are able to show that this is so for terms in canonical form.



(3.1)

Alternatively one could also use SLC_α (see Section 3.5 below for the definition) and just work with the term in canonical form (we can use the same proof technique). All our proofs are such that either interpretation works, i.e the reader may always replace SLC by SLC_α and imagine that we pick the representative in canonical form.

3.5 Action on Morphisms

We define $\mathrm{SLC}\, f : \mathrm{SLC}\, X \to \mathrm{SLC}\, Y$ for all morphisms $f : X \to Y$ in **Set** by $\mathrm{SLC}\, f \coloneqq \overline{f_X} \circ \mathrm{C}_X$, where $\overline{f_X} : \mathrm{SLC}\, X \to \mathrm{SLC}\, Y$ is defined recursively for all $X \in \mathbf{Set}$ and all $L \in \mathrm{SLC}\, X$ by

$$L \mapsto \begin{cases} \mathrm{var}_Y(f(x)), & \text{if } L = \mathrm{var}_X(x) \text{ with } x \in X, \\ \mathrm{app}_Y(\overline{f_X}(L_1), \overline{f_X}(L_2)), & \text{if } L = \mathrm{app}_X(L_1, L_2) \text{ with } L_1, L_2 \in \mathrm{SLC}\, X, \\ \mathrm{abs}_Y(\overline{f_{X^*}}(L')), & \text{if } L = \mathrm{abs}_X(L') \text{ with } L' \in \mathrm{SLC}'\, X, \\ \lambda c.c, & \text{otherwise,} \end{cases}$$

where $c \coloneqq \min\{d \mid d \in D\}$. In the second line we used that if $L = \mathrm{app}_X(L_1, L_2)$ is in canonical form so are L_1 and L_2. Note that by the remark before Lemma 3.1 we always land in one of the first three cases due to precomposition with C_X.

This construction satisfies $\mathrm{SLC}\, \mathrm{Id}_X = \mathrm{Id}_{\mathrm{SLC}\, X}$ and $\mathrm{SLC}(h \circ g) = \mathrm{SLC}\, h \circ \mathrm{SLC}\, g$ for all $X \in \mathbf{Set}$ and all pairs of composable morphisms h and g.

For $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$ we denote by SLC_δ the construction similar to $\mathrm{SLC}\, X$ which sends every $X \in \mathbf{Set}$ to $\mathrm{SLC}\, X$ modulo the equivalence relation generated by δ-conversions on λ-terms. For this to be well-defined we just need to note that $\mathrm{SLC}\, f$ for any morphism $f : X \to Y$ in **Set** is constant on equivalence classes.

Lemma 3.2 $\mathrm{SLC}\, f : \mathrm{SLC}\, X \to \mathrm{SLC}\, Y$ *is constant on equivalence classes of* SLC_δ *for all morphisms* $f : X \to Y$ *in* **Set** *and all* $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$.

Proof Let $f\colon X \to Y$ be any morphism in **Set**. We show that $\mathrm{SLC}\, f$ is constant with respect to any of the three equivalence relations separately. This implies all four statements.

$\mathrm{SLC}\, f$ is clearly constant with respect to α-equivalence due to the precomposition with C_X and the fact that α-equivalent λ-terms have the same canonical form.

For β- and η-equivalence it is enough to show that two λ-terms differing by one such step are equivalent – the statement then follows by induction. Further the recursive definition of $\mathrm{SLC}\, f$ shows that this is enough to show that two λ-terms containing equivalent subterms (and being equal otherwise) get mapped to the same equivalence class.

First consider β-equivalence: Assume that $L \in \mathrm{SLC}\, X$ contains a β-redex, i.e. is of the form $\mathrm{app}_X(\mathrm{abs}_X(L_1), L_2)$ with $L_1 \in \mathrm{SLC}'\, X$ and $L_2 \in \mathrm{SLC}\, X$. We then get the following chain of equalities:

$$\begin{aligned}
\mathrm{SLC}\, f(\mathrm{app}_X(\mathrm{abs}_X(L_1), L_2)) &= \mathrm{app}_Y(\mathrm{SLC}\, f(\mathrm{abs}_X(L_1)), \mathrm{SLC}\, f(L_2)) \\
&= \mathrm{app}_Y(\mathrm{abs}_Y(\mathrm{SLC}\, f^*(L_1)), \mathrm{SLC}\, f(L_2)) \\
&= \mathrm{SLC}\, f^*(L_1)[* \leftarrow \mathrm{SLC}\, f(L_2)] \\
&= \mathrm{SLC}\, f(L_1[* \leftarrow L_2]).
\end{aligned}$$

The first two equalities hold by the definition of $\mathrm{SLC}\, f$ and the third one by β-equivalence (on $\mathrm{SLC}\, Y$). The last equality holds by showing the outer rectangle in

$$\begin{array}{ccccc}
\mathrm{SLC}\, X^* & \xrightarrow{\mathrm{SLC}\, \gamma_1} & \mathrm{SLC}\, \mathrm{SLC}\, X & \xrightarrow{\mu_X^{(\mathrm{SLC})}} & \mathrm{SLC}\, X \\
\downarrow{\scriptstyle \mathrm{SLC}\, f^*} & & \downarrow{\scriptstyle \mathrm{SLC}\, \mathrm{SLC}\, f} & & \downarrow{\scriptstyle \mathrm{SLC}\, f} \\
\mathrm{SLC}\, Y^* & \xrightarrow[\mathrm{SLC}\, \gamma_2]{} & \mathrm{SLC}\, \mathrm{SLC}\, Y & \xrightarrow[\mu_Y^{(\mathrm{SLC})}]{} & \mathrm{SLC}\, Y
\end{array}$$

commutative, where $\gamma_1\colon X^* \to \mathrm{SLC}\, X$ is given for all $z \in X^*$ by

$$z \mapsto \begin{cases} \eta_X^{(\mathrm{SLC})}(z), & \text{if } z \in X, \\ L_2, & \text{if } z = *. \end{cases}$$

and $\gamma_2\colon Y^* \to \mathrm{SLC}\, Y$ is given for all $z \in Y$ by

$$z \mapsto \begin{cases} \eta_Y^{(\mathrm{SLC})}(z), & \text{if } z \in Y, \\ \mathrm{SLC}\, f(L_2), & \text{if } z = *. \end{cases}$$

In this diagram the right square commutes as $\mu^{(\mathrm{SLC})}$ is a natural transformation (cf. proof of Lemma 3.3) and the left square commutes as SLC is a functor and

the square

$$\begin{array}{ccc} X^* & \xrightarrow{\gamma_1} & \mathrm{SLC}\, X \\ {\scriptstyle f^*}\downarrow & & \downarrow{\scriptstyle \mathrm{SLC}\, f} \\ Y^* & \xrightarrow[\gamma_2]{} & \mathrm{SLC}\, Y \end{array}$$

is clearly commutative.

It is left to show that $\mathrm{SLC}\, f$ is constant with respect to η-equivalence. Therefore assume that $L \in \mathrm{SLC}\, X$ is a λ-term containing an η-redex, i.e. L is of the form $\mathrm{abs}_X(\mathrm{app}_{X^*}(\mathrm{SLC}\, \iota_X(K), \mathrm{var}_{X^*}(*)))$ with $K \in \mathrm{SLC}\, X$. We then get the following chain of equalities:

$$\begin{aligned} &\mathrm{SLC}\, f(\mathrm{abs}_X(\mathrm{app}_{X^*}(\mathrm{SLC}\, \iota_X(K), \mathrm{var}_{X^*}(*)))) \\ &\qquad = \mathrm{abs}_Y(\mathrm{SLC}\, f^*(\mathrm{app}_{X^*}(\iota_X(K), \mathrm{var}_{X^*}(*)))) \\ &\qquad = \mathrm{abs}_Y(\mathrm{app}_{Y^*}(\mathrm{SLC}\, f^*(\iota_X(K)), \mathrm{SLC}\, f^*(\mathrm{var}_{X^*}(*)))) \\ &\qquad = \mathrm{abs}_Y(\mathrm{app}_{Y^*}(\mathrm{SLC}\, f^*(\iota_X(K)), \mathrm{var}_{Y^*}(f^*(*)))) \\ &\qquad = \mathrm{abs}_Y(\mathrm{app}_{Y^*}(\mathrm{SLC}\, f^*(\iota_X(K)), \mathrm{var}_{Y^*}(*))) \\ &\qquad = \mathrm{abs}_Y(\mathrm{app}_{Y^*}(\iota_Y(\mathrm{SLC}\, f(K)), \mathrm{var}_{Y^*}(*))) \\ &\qquad = \mathrm{SLC}\, f(K). \end{aligned}$$

Here the first four equations hold by the definition of $\mathrm{SLC}\, f$ and the last one holds by η-equivalence (on $\mathrm{SLC}\, Y$). For the fifth equation note that

$$\begin{array}{ccc} \mathrm{SLC}\, X & \xrightarrow{\mathrm{SLC}\, \iota_X} & \mathrm{SLC}\, X^* \\ {\scriptstyle \mathrm{SLC}\, f}\downarrow & & \downarrow{\scriptstyle \mathrm{SLC}\, f^*} \\ \mathrm{SLC}\, Y & \xrightarrow[\mathrm{SLC}\, \iota_Y]{} & \mathrm{SLC}\, Y^* \end{array}$$

commutes, as SLC is a functor and the square

$$\begin{array}{ccc} X & \xrightarrow{\iota_X} & X^* \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle f^*} \\ Y & \xrightarrow[\iota_Y]{} & Y^* \end{array}$$

obviously commutes. □

We write LC for $\mathrm{SLC}_{\alpha\beta\eta}$. This is our interpretation of lambda calculus, hence the name LC. For all $X \in \mathbf{Set}$ and all $L \in \mathrm{SLC}\, X$ we denote by $\overline{L}$ its equivalence class in $\mathrm{SLC}_\delta\, X$ for all $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$. For an arbitrary equivalence class we usually use the letter T and the letter L for elements in $\mathrm{SLC}\, X$ therein. If it

is clear which functor we are dealing with we omit the upper indices for var, app, and abs.

As α-equivalent λ-terms have the same canonical form, $\deg\colon \mathrm{SLC}\,X \to \mathbb{N}$ is constant on equivalence classes with respect to α-equivalence. However this is obviously not the case for β- and η-equivalences respectively. We therefore define $\deg\colon \mathrm{SLC}_\delta \to \mathbb{N}$ for all $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$, all $X \in \mathbf{Set}$, and all $T \in \mathrm{SLC}_\delta\, X$ by

$$\deg(T) \coloneqq \min\{\deg(L) \mid L \in \mathrm{SLC}\,X \text{ with } L \in T\}.$$

This is well-defined as equivalence classes are not empty and picking any representative in T gives an upper bound for $\deg(T)$. We are not concerned how one can actually compute the degree of a λ-term.

3.6 (Exponential) Monad Structure

We define two maps on SLC that induce maps on SLC_δ that satisfy the conditions of the unit and multiplication of a monad for $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$. However (as explained further after Lemma 3.3) these maps will *not* be a unit and a multiplication for SLC itself.

The maps $\eta_X^{(\mathrm{SLC})}\colon X \to \mathrm{SLC}\,X$ and $\mu_X^{(\mathrm{SLC})}\colon \mathrm{SLC}\,\mathrm{SLC}\,X \to \mathrm{SLC}\,X$ are given for all $X \in \mathbf{Set}$ by $\eta_X^{(\mathrm{SLC})} \coloneqq \mathrm{var}_X$ (i.e. insertion of variables) and by letting $\mu_X^{(\mathrm{SLC})}$ be replacement (i.e. a variable corresponding to a λ-term gets replaced by its corresponding λ-term). More precisely replacement is given by first mapping any term to its canonical form, then replacing the free variables by their corresponding λ-terms, and finally identifying all bound variables with the "same" bound variables (according to the respective inclusions in the disjoint union). Remember that this function then gets composed with C_X. It is not hard to see that these maps are constant on equivalence classes with respect to α-, β-, and η-equivalence.

Lemma 3.3 *For $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$ the triples $(\mathrm{SLC}_\delta, \eta^{(\mathrm{SLC}_\delta)}, \mu^{(\mathrm{SLC}_\delta)})$ define a monad.*

Proof It is easy to see that the diagrams corresponding to (2.1) commute for the triples $(\mathrm{SLC}_\delta, \eta^{(\mathrm{SLC}_\delta)}, \mu^{(\mathrm{SLC}_\delta)})$ with $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$. The commutativity of the diagrams corresponding to (2.2) follows from the substitution lemma of lambda calculus (cf. [Bar84, Lemma 2.1.16] for the statement and a proof of that lemma). Hence it is left to show naturality.

As the maps $\eta^{(\mathrm{SLC}_\delta)}$ and $\mu^{(\mathrm{SLC}_\delta)}$ come from maps for SLC we will prove all statements (if possible) in that setting as this implies the corresponding statements for SLC_δ.

It is easy to see that $\eta^{(\mathrm{SLC})}$ is a natural transformation. For $\mu^{(\mathrm{SLC})}$ we prove naturality by induction on the degree of the λ-terms. Therefore let $f\colon X \to Y$

be any morphism in **Set**. We need to prove

$$\begin{array}{ccc} \mathrm{SLC}\,\mathrm{SLC}\,X & \xrightarrow{\mu_X^{(\mathrm{SLC})}} & \mathrm{SLC}\,X \\ \downarrow{\scriptstyle \mathrm{SLC}\,\mathrm{SLC}\,f} & & \downarrow{\scriptstyle \mathrm{SLC}\,f} \\ \mathrm{SLC}\,\mathrm{SLC}\,Y & \xrightarrow[\mu_Y^{(\mathrm{SLC})}]{} & \mathrm{SLC}\,Y \end{array}$$

commutative. We use induction on the outer degree, therefore the base case is for λ-terms in $\mathrm{SLC}^{\langle 0\rangle}\,\mathrm{SLC}\,X$. For any $L \in \mathrm{SLC}\,X$ we denote by $x_L \in \mathrm{SLC}\,\mathrm{SLC}\,X$ the variable corresponding to L, i.e. we have $\mu_X^{(\mathrm{SLC})}(x_L) = \mu_X^{(\mathrm{SLC})}(\mathrm{var}_{\mathrm{SLC}\,X}(L)) = L$. Therefore let $L \in \mathrm{SLC}^{\langle 0\rangle}\,\mathrm{SLC}\,X$ be arbitrary. The base case then holds by

$$\mathrm{SLC}\,f(\mu_X^{(\mathrm{SLC})}(x_L)) = \mathrm{SLC}\,f(L) = \mu_Y^{(\mathrm{SLC})}(x_{\mathrm{SLC}\,f(L)}) = \mu_Y^{(\mathrm{SLC})}(\mathrm{SLC}\,\mathrm{SLC}\,f(L)).$$

If $L = \mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2) \in \mathrm{SLC}\,\mathrm{SLC}\,X$ with $L_1, L_2 \in \mathrm{SLC}\,\mathrm{SLC}\,X$ we have the following two induction hypotheses:

$$\mathrm{SLC}\,f(\mu_X^{(\mathrm{SLC})}(L_i)) = \mu_Y^{(\mathrm{SLC})}(\mathrm{SLC}\,\mathrm{SLC}\,f(L_i))$$

for $i \in \{1, 2\}$. The statement we need to show is

$$\mathrm{SLC}\,f(\mu_X^{(\mathrm{SLC})}(\mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2))) = \mu_Y^{(\mathrm{SLC})}(\mathrm{SLC}\,\mathrm{SLC}\,f(\mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2))).$$

As application and multiplication commute by the definition of lambda calculus and application and $\mathrm{SLC}\,f$ commute by the definition of SLC we get

$$\begin{aligned} &\mathrm{SLC}\,f(\mu_X^{(\mathrm{SLC})}(\mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2))) \\ &\qquad = \mathrm{SLC}\,f(\mathrm{app}_X(\mu_X^{(\mathrm{SLC})}(L_1), \mu_X^{(\mathrm{SLC})}(L_2))) \\ &\qquad = \mathrm{app}_Y(\mathrm{SLC}\,f(\mu_X^{(\mathrm{SLC})}(L_1)), \mathrm{SLC}\,f(\mu_X^{(\mathrm{SLC})}(L_1))). \end{aligned}$$

Then, using the induction hypotheses and the same arguments as above, the equalities

$$\begin{aligned} &\mathrm{app}_Y(\mathrm{SLC}\,f(\mu_X^{(\mathrm{SLC})}(L_1)), \mathrm{SLC}\,f(\mu_X^{(\mathrm{SLC})}(L_1))) \\ &\qquad = \mathrm{app}_Y(\mu_Y^{(\mathrm{SLC})}(\mathrm{SLC}\,\mathrm{SLC}\,f(L_1)), \mu_Y^{(\mathrm{SLC})}(\mathrm{SLC}\,\mathrm{SLC}\,f(L_2))) \\ &\qquad = \mu_Y^{(\mathrm{SLC})}(\mathrm{app}_{\mathrm{SLC}\,Y}(\mathrm{SLC}\,\mathrm{SLC}\,f(L_1), \mathrm{SLC}\,\mathrm{SLC}\,f(L_2))) \\ &\qquad = \mu_Y^{(\mathrm{SLC})}(\mathrm{SLC}\,\mathrm{SLC}\,f(\mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2))) \end{aligned}$$

hold, which shows the first inductive step.

If $L = \mathrm{abs}_{\mathrm{SLC}\,X}(L') \in \mathrm{SLC}\,\mathrm{SLC}\,X$ with $L' \in \mathrm{SLC}'\,\mathrm{SLC}\,X$ we have the following induction hypothesis:

$$\mathrm{SLC}'\,f(\mu_{X^*}^{(\mathrm{SLC})}(\mathrm{SLC}\,g_X(L'))) = \mu_{Y^*}^{(\mathrm{SLC})}(\mathrm{SLC}\,\mathrm{SLC}'\,f(\mathrm{SLC}\,g_X(L'))),$$

where the map $g_X : (\mathrm{SLC}\, X)^* \to \mathrm{SLC}\, X^*$ is defined by $g_X|_{\mathrm{SLC}\, X} = \mathrm{SLC}\, \iota_X$ and $g_X(*) = \eta^{(\mathrm{SLC})}_{X^*}(*)$. Note that g_X is an injection satisfying the condition $\deg(L') = \deg(\mathrm{SLC}\, g_X(L'))$. We then need to show that

$$\mathrm{SLC}\, f(\mu^{(\mathrm{SLC})}_X(\mathrm{abs}_{\mathrm{SLC}\, X}(L'))) = \mu^{(\mathrm{SLC})}_Y(\mathrm{SLC}\,\mathrm{SLC}\, f(\mathrm{abs}_{\mathrm{SLC}\, X}(L')))$$

holds. The proof can be visualized in the diagram

$$\begin{array}{ccccc}
\mathrm{SLC}'\,\mathrm{SLC}\, X & \xrightarrow{\mathrm{SLC}\, g_X} & \mathrm{SLC}\,\mathrm{SLC}'\, X & \xrightarrow{\mu^{(\mathrm{SLC})}_{X^*}} & \mathrm{SLC}\, X \\
\downarrow{\scriptstyle \mathrm{abs}_{\mathrm{SLC}\, X}} & & & & \downarrow{\scriptstyle \mathrm{abs}_X} \\
\mathrm{SLC}\,\mathrm{SLC}\, X & & \xrightarrow{\mu^{(\mathrm{SLC})}_X} & & \mathrm{SLC}\, X \\
\downarrow{\scriptstyle \mathrm{SLC}\,\mathrm{SLC}\, f} & & & & \downarrow{\scriptstyle \mathrm{SLC}\, f} \\
\mathrm{SLC}\,\mathrm{SLC}\, Y & & \xrightarrow[\mu^{(\mathrm{SLC})}_Y]{} & & \mathrm{SLC}\, Y,
\end{array} \tag{3.2}$$

where the upper rectangle commutes by the definition of $\mu^{(\mathrm{SLC})}$ and the definition of lambda calculus.

Let $L' \in \mathrm{SLC}'\,\mathrm{SLC}\, X$ be arbitrary. We get the following chain of equalities:

$$\begin{aligned}
&\mathrm{SLC}\, f(\mu^{(\mathrm{SLC})}_X(\mathrm{abs}_{\mathrm{SLC}\, X}(L'))) \\
&\qquad = \mathrm{SLC}\, f(\mathrm{abs}_X(\mu^{(\mathrm{SLC})}_{X^*}(\mathrm{SLC}\, g_X(L')))) && (3.3)\\
&\qquad = \mathrm{abs}_Y(\mathrm{SLC}'\, f(\mu^{(\mathrm{SLC})}_{X^*}(\mathrm{SLC}\, g_X(L')))) && (3.4)\\
&\qquad = \mathrm{abs}_Y(\mu^{(\mathrm{SLC})}_{Y^*}(\mathrm{SLC}\,\mathrm{SLC}'\, f(\mathrm{SLC}\, g_X(L')))) && (3.5)\\
&\qquad = \mathrm{abs}_Y(\mu^{(\mathrm{SLC})}_{Y^*}(\mathrm{SLC}\, g_X(\mathrm{SLC}'\,\mathrm{SLC}\, f(L')))) && (3.6)\\
&\qquad = \mu^{(\mathrm{SLC})}_Y(\mathrm{abs}_{\mathrm{SLC}\, Y}(\mathrm{SLC}'\,\mathrm{SLC}\, f(L'))) && (3.7)\\
&\qquad = \mu^{(\mathrm{SLC})}_Y(\mathrm{SLC}\,\mathrm{SLC}\, f(\mathrm{abs}_{\mathrm{SLC}\, X}(L'))). && (3.8)
\end{aligned}$$

We now give arguments why these equations hold. (3.3) and (3.7) hold by the commutativity of the upper rectangle in (3.2) (in the second case with X replaced by Y). (3.4) and (3.8) are just the definitions of the action of SLC on morphisms. Further (3.5) is the induction hypothesis. Hence it is left to show that (3.6) holds. This follows from the commutativity of

$$\begin{array}{ccc}
\mathrm{SLC}(\mathrm{SLC}(X) \amalg *) & \xrightarrow{\mathrm{SLC}\, g_X} & \mathrm{SLC}(\mathrm{SLC}(X \amalg *)) \\
\downarrow{\scriptstyle \mathrm{SLC}(\mathrm{SLC}\, f)^*} & & \downarrow{\scriptstyle \mathrm{SLC}(\mathrm{SLC}\, f^*)} \\
\mathrm{SLC}(\mathrm{SLC}(Y) \amalg *) & \xrightarrow[\mathrm{SLC}\, g_Y]{} & \mathrm{SLC}(\mathrm{SLC}(Y \amalg *)),
\end{array}$$

which commutes as SLC is a functor and

$$\begin{array}{ccc} \mathrm{SLC}(X) \amalg * & \xrightarrow{g_X} & \mathrm{SLC}(X \amalg *) \\ \downarrow{\scriptstyle (\mathrm{SLC}\, f)^*} & & \downarrow{\scriptstyle \mathrm{SLC}\, f^*} \\ \mathrm{SLC}(Y) \amalg * & \xrightarrow[g_Y]{} & \mathrm{SLC}(Y \amalg *), \end{array}$$

is clearly commutative. This finishes the second case of the induction and hence proves that $\mu^{(\mathrm{SLC})}$ is actually a natural transformation. □

Note that the triple $(\mathrm{SLC}, \eta^{(\mathrm{SLC})}, \mu^{(\mathrm{SLC})})$ fails to satisfy the commutativity of the diagram corresponding to (2.1) by α-equivalence (start with a λ-term that is not in canonical form). This could be avoided by (properly) defining replacement without invoking canonical forms. However for the naturality of this replacement we would then need to change the action of SLC on morphisms to one that is not constant with respect to δ-equivalence for $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$. Therefore SLC can not be turned into a monad in a reasonable way.

Finally we define the exponential structure on LC. We use the fact that abs is a morphism of LC-modules which is shown in the next section. For all $X \in \mathbf{Set}$ we define a map $\mathrm{app1}_X \colon \mathrm{SLC}\, X \to \mathrm{SLC}'\, X$ for all $L \in \mathrm{SLC}\, X$ by

$$\mathrm{app1}_X(L) \coloneqq \mathrm{app}_{X^*}(\mathrm{SLC}\, \iota_X(L), \mathrm{var}_{X^*}(*)).$$

By [Zsi06, Remark 2.23] this defines a morphism $\mathrm{app1} \colon \mathrm{LC} \to \mathrm{LC}'$ of LC-modules. Further by setting $\varepsilon^{(\mathrm{LC})} = \mathrm{app1}$ and $\varepsilon^{(\mathrm{LC})^{-1}} = \mathrm{abs}$ we see that LC is an exponential monad as for all $X \in \mathbf{Set}$ and all $T' \in \mathrm{LC}'\, X$ we have $\mathrm{app1}_X(\mathrm{abs}_X(T')) = T'$ by β-equivalence and for all $T \in \mathrm{LC}\, X$ we get $\mathrm{abs}_X(\mathrm{app1}_X(T)) = T$ by η-equivalence.

3.7 Morphisms of Lambda Calculus

Our goal is to show that basic operations of lambda calculus can be seen as maps or even morphisms in our setting. We already used the operation var of lambda calculus for the unit of our monads. Further we can regard app as a morphism of LC-modules $\mathrm{LC} \times \mathrm{LC} \to \mathrm{LC}$ and abs as a morphism of LC-modules $\mathrm{LC}' \to \mathrm{LC}$:

Lemma 3.4 $\mathrm{app} \colon \mathrm{LC} \times \mathrm{LC} \to \mathrm{LC}$ *and* $\mathrm{abs} \colon \mathrm{LC}' \to \mathrm{LC}$ *are morphisms of* LC*-modules.*

Proof The naturality of app and abs means that they commute with morphisms induced by morphisms in **Set**. This is satisfied by the definition of the action of SLC on any morphism $f \colon X \to Y$ in **Set**.

app and abs turn the diagrams corresponding to (2.6) commutative by the definition of substitution (cf. Examples 2.14 and 2.15 in [Zsi06]). □

Further we can use replacement (i.e. $\mu^{(\mathrm{LC})}$) to describe substitution as already used in Lemma 3.2: Given $T_1, T_2 \in \mathrm{LC}\, X$ for some $X \in \mathbf{Set}$ we get $T_1[x \leftarrow T_2]$ for $x \in X$ by chasing $T_1 \in \mathrm{LC}\, X$ through the following chain of maps:

$$\mathrm{LC}\, X \xrightarrow{\mathrm{LC}\,\gamma} \mathrm{LC}\,\mathrm{LC}\, X \xrightarrow{\mu_X^{(\mathrm{LC})}} \mathrm{LC}\, X, \tag{3.9}$$

where $\gamma\colon X \to \mathrm{LC}\, X$ is given for all $y \in X$ by

$$y \mapsto \begin{cases} \overline{\mathrm{var}_X^{(\mathrm{SLC})}(y)} = \eta_X^{(\mathrm{LC})}(y), & \text{if } x \neq y \in X, \\ T_2, & \text{if } y = x. \end{cases}$$

Note that we can consider T_2 as an equivalence class in $\mathrm{LC}(X \setminus x)$ if $x \notin \mathrm{FV}(T_2)$. We can therefore regard γ as a map from X to $\mathrm{LC}(X \setminus x)$ (this is later used for $X = Y^*$, where $Y \in \mathbf{Set}$ and $x = *$).

3.8 Justification of the Definition

As stated earlier we now need to justify why our definition of LC is equivalent to "the" standard definition of lambda calculus. Here we need to pay attention that different authors use different definitions for lambda calculus. All the usual definitions take some variables and then build all λ-terms involving those variables. Hence the main difference to our definition is that there is no set of variables that can only occur bound. Where the different definitions differ is in what the variables should be. For example [Bar84] assumes that the variables form a countably infinite set but [Sel07] just assumes that the variables form any infinite set.We can now justify our definition as follows:

(i) By the example at the beginning of this section we see that "usual" lambda calculus is not well-behaved under renaming variables, i.e. it is not functorial in general. Hence there is some change needed such that it is "well-behaved".

(ii) "Usual" lambda calculus is not defined on non-empty, finite sets: Assume we form all λ-terms with variables in the set $X = \{x_1, x_2, \ldots, x_m\}$ with $m \in \mathbb{N}$ and $m \geq 1$. We then consider the λ-term

$$\lambda x_2.(\lambda x_1.x_1 \cdots x_m)(x_1 \cdots x_m).$$

By β-equivalence we get a λ-term of the form

$$(\lambda x_1.x_1 \cdots x_m)[x_2 \leftarrow (x_1 \cdots x_m)].$$

However this λ-term is not defined as $x_1 \in \mathrm{FV}(x_1 \cdots x_m)$ and we can not apply an α-conversion without changing the meaning of the first λ-term (and we still couldn't apply substitution for the same reason). Additionally the author is not aware why one should consider lambda calculus on an empty set of variables (as it would be the empty set itself).

(iii) Adding only a single variable that needs to be bound would already eliminate this problem, however a similar argument as the example at the beginning of this section would show that the "renaming morphism" would still not be well-behaved in general. We therefore need a place to "store" or "hide" the bound variables from renaming. Note that if X is infinite, a λ-term over X may have arbitrarily many different bound variables. This has the following two consequences: First the set of strictly bound variables for X needs to be infinite as to be able to "hide" all bound variables of all possible λ-terms over X. Second if we want to map these variables injectively into bound variables of another (possibly finite) set we need to have arbitrarily many bound variables for this set too, i.e. this set needs to be infinite.

(iv) Another (although weaker) argument to take infinitely many bound variables is the following: If we would only add a finite set of bound variables, app1 and abs would not form an isomorphism for finite sets (using the pigeonhole principle): Assume we form all λ-terms with free variables in a finite, non-empty set $X = \{x_1, x_2, \ldots, x_m\}$ and bound variables in a finite (possibly empty) set $Y = \{y_1, y_2, \ldots, y_n\}$ with $X \cap Y = \emptyset$. We then consider the λ-term

$$\lambda x_1 \cdots x_m y_1 \cdots y_n . x_1 \cdots x_m y_1 \cdots y_n *$$

over the variables $X^* \cup Y$. Applying abs_X yields

$$\lambda * . (\lambda x_1 \cdots x_m y_1 \cdots y_n . x_1 \cdots x_m y_1 \cdots y_n *),$$

but as $* \notin (X \cup Y)$ we need to replace it by an element in $X \cup Y$. No matter which one we pick, $\mathrm{app}\, 1_X$ can not return the λ-term we started with.

(v) By Lemma 3.5 below it does not matter what infinite set of variables we take, hence there is no argument not to take the smallest possible. We might even take different sets of strictly bound variables (not necessarily of the same cardinality) for different sets. The statements used here would still hold with the same ideas but it would be harder to formulate proofs.

(vi) For infinite sets $X \in \mathbf{Set}$ and *injective* maps between these sets we get by Lemma 3.6 that our definition coincides with the "usual" one. The outlined proof thereof also shows how one could (with a simple change) extend this version to any maps between infinite sets.

(vii) One might be tempted to take SLC as the endofunctor that assigns to each $X \in \mathbf{Set}$ all λ-terms with free variables in X and arbitrary bound variables, i.e. the set of all variables would be

$$\mathrm{VAR} := \bigcup_{X \in \mathbf{Set}} \{x \mid x \in X\}.$$

However this union would form a proper class, i.e. the codomain of SLC would not be **Set**. This would make notation easier, but most definitions

impossible: First we would need to extend lambda calculus to classes, then the λ-terms formed over a set could not be regarded as elements of a set (otherwise we would have that all λ-terms are variables), and finally even after equivalence relations on classes we would not get sets, i.e. no equivalence to "usual" lambda calculus.

(viii) After the previous points we have two possible ways we can take: Either we just consider SLC and SLC_δ for $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$ as endofunctors on $\mathbf{Set}_{\mathrm{inj}}^{\infty}$, the category of infinite sets with injective morphisms (where "usual" lambda calculus applies), or we can extend lambda calculus in a (hopefully) reasonable (and minimal) way to finite sets and arbitrary morphisms. The first possibility has the advantage of precisely capturing lambda calculus but the second one has the advantage that the endofunctors are over a category with many (nice) properties. The proof of the main theorem involves one step (the generalization of application in Section 4.2) that uses an additional property of the category **Set** which does *not* hold in $\mathbf{Set}_{\mathrm{inj}}^{\infty}$. Hence our proof does not work in that setting. However if we use the remark in the proof of Lemma 3.6 and work in the category of infinite sets with all morphisms between these sets the proof still works, i.e. the reader may still derive the result in that setting if needed.

Lemma 3.5 *For all $X \in \mathbf{Set}$ there is a natural isomorphism $\mathrm{SLC}_\delta\, X \cong \widetilde{\mathrm{SLC}_\delta} X$ for $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$, where $\widetilde{\mathrm{SLC}_\delta} : \mathbf{Set} \to \mathbf{Set}$ are the functors defined in a similar way as SLC_δ but with the set D replaced by any infinite set E.*

Proof (Sketch) A complete proof of this statement would require a proof that all maps involved are well-defined which we omit as they are (almost) the same as for SLC_δ. However it is important to note that we do not impose any ordering of the elements of the set E. Hence the canonical form described below contains a choice which means that $\widetilde{\mathrm{SLC}}$ is not well-defined. However the monads $\widetilde{\mathrm{SLC}_\delta}$ for $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$ are well-defined as they all contain α-equivalence.

It is enough to proof the statement for $\delta = \alpha$ as this implies (by construction) the other results. First fix a countably infinite subset $F \subseteq E$ and equip it with a total order such as D. We can then define a map $\varphi : D \to F$ which sends the smallest element of D to the smallest one of F and recursively the successor of any element of D to the successor of the corresponding predecessor in F. This map is clearly an isomorphism of totally ordered sets. Further we define a canonical representative for all $X \in \mathbf{Set}$ and all $L \in \widetilde{\mathrm{SLC}} X$ with respect to F in a similar way as we did for SLC and D. This representative is denoted by $\widetilde{\mathrm{C}}_X(L)$.

We then define two maps $\xi_X : \mathrm{SLC}\, X \to \widetilde{\mathrm{SLC}} X$ and $\zeta_X : \widetilde{\mathrm{SLC}} X \to \mathrm{SLC}\, X$ for all $X \in \mathbf{Set}$, all $L \in \mathrm{SLC}\, X$, and all $K \in \widetilde{\mathrm{SLC}} X$ by

$$\xi_X(L) \coloneqq \widetilde{\varphi}((L)_{\mathrm{C}_X}),$$
$$\zeta_X(K) \coloneqq \widetilde{\varphi^{-1}}((K)_{\widetilde{\mathrm{C}}_X}),$$

where $\widetilde{\varphi}$ is the map that exchanges all bound variables in $(L)_{C_X}$ according to φ (they all belong to D by the definition of $(L)_{C_X}$) and similarly for $\widetilde{\varphi^{-1}}$.

Note that none of the involved maps changes the "structure" of a λ-term, all changes can be achieved by α-conversions (in a suitable set of bound variables). It is now easy to see that ξ and ζ induce an isomorphism between SLC_α and $\widetilde{\mathrm{SLC}_\alpha}$ (any two α-equivalent λ-terms have the same canonical form and ξ and ζ induce a bijection of canonical forms). Naturality of this isomorphism follows by the fact that the actions on morphisms are defined in a similar way, just with respect to to different canonical forms. Similarly it could be shown that this actually defines a natural isomorphism of exponential monads (in the case of LC and $\widetilde{\mathrm{LC}}$ respectively). □

Lemma 3.6 *We define a functor* $\widehat{\mathrm{SLC}}\colon \mathbf{Set}^\infty_{\mathrm{inj}} \to \mathbf{Set}^\infty_{\mathrm{inj}}$ *for all* $X \in \mathbf{Set}^\infty_{\mathrm{inj}}$ *by*

$$X \mapsto \{L \mid L \text{ is a } \lambda\text{-term with variables in } X\}$$

and for all (injective) morphisms $f\colon X \to Y$ *in* $\mathbf{Set}^\infty_{\mathrm{inj}}$ *for all* $L \in \widehat{\mathrm{SLC}}X$ *recursively by*

$$L \mapsto \begin{cases} \mathrm{var}_Y(f(x)), & \text{if } L = \mathrm{var}_X(x) \text{ with } x \in X, \\ \mathrm{app}_Y(\widehat{\mathrm{SLC}}f(L_1), \widehat{\mathrm{SLC}}f(L_2)), & \text{if } L = \mathrm{app}_X(L_1, L_2) \\ & \text{with } L_1, L_2 \in \widehat{\mathrm{SLC}}X, \\ \lambda\, \mathrm{var}_Y(f(x)).(\widehat{\mathrm{SLC}}(f)(L')), & \text{if } L = \lambda x.L' \\ & \text{with } x \in X \text{ and } L' \in \widehat{\mathrm{SLC}}X. \end{cases}$$

One can define $\widehat{\mathrm{SLC}_\delta}$ *similarly to* SLC_δ *for* $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$.

We claim that there is a natural isomorphism $\mathrm{SLC}_\delta\, Z \cong \widehat{\mathrm{SLC}_\delta} Z$ *for all infinite sets* $Z \in \mathbf{Set}$ *and injective maps between them, i.e. there is a natural isomorphism between* SLC_δ *and* $\widehat{\mathrm{SLC}_\delta}$ *if we restrict domain and codomain of* SLC_δ *to* $\mathbf{Set}^\infty_{\mathrm{inj}}$.

Proof (Sketch) We again only give a sketch of how such a proof works, similar to Lemma 3.5 above. Before we give the sketch of the proof there are two points one should note: The first one is that $\widehat{\mathrm{SLC}}$ is actually an endofunctor, i.e. it does map injective morphisms between sets to injective morphisms. The second point is that as above the canonical form of λ-terms used in this proof is only well-defined after applying α-equivalence.

As in Lemma 3.5 it is enough to prove the statement for $\delta = \alpha$. The idea is the same as in that lemma, it just involves a different canonical form.

For all $X \in \mathbf{Set}^\infty_{\mathrm{inj}}$ fix a countably infinite subset $\widehat{X} \subseteq X$ and equip it with a total order such as D. For simplicity we assume that $\widehat{X}$ is given by $\{\widehat{x_j} \mid j \in \mathbb{N}\}$. For the canonical representative we need a small change as an arbitrary λ-term $L \in \widehat{\mathrm{SLC}}X$ may contain "small" variables (with respect to the corresponding total order) of $\widehat{X}$ as free variables. Therefore we need to change the second stage of the algorithm in Section 3.9 in the following way:

For every edge that is labelled with a "0" the algorithm searches the highest label of any edge in the subtree starting with the source node of this edge. Let this number be $n \in \mathbb{N}$. Then define the subset $\widehat{X_n} \subseteq \widehat{X}$ as the (in ascending order) smallest $n+1$ elements in $\widehat{X}$ which are not free in this subtree (note that this set is dependent on the subtree of the considered λ-term and not just on the natural number n – we just don't want to introduce too many indices). Such a set always exists as $\widehat{X}$ is infinite and any λ-term can only contain a finite number of free (and bound) variables. For notational convenience we assume that $\widehat{X_n}$ is given by $\{\widehat{y_j} \mid 0 \leq j \leq n\}$. Every labelled variable is in exactly one of these subtrees. If the variable is labelled with $i \in \mathbb{N}$, it is replaced by $\widehat{y_k}$ with $k = n - i$. The relabelling of the edges is given as follows: An edge labelled with $i \in \mathbb{N}$ is relabelled with the index of $\widehat{y_k}$ with $k = n - i$ inside $\widehat{X}$ (e.g. if $\widehat{y_k} = \widehat{x_l}$, then the edge is labelled with "l"). The algorithm then goes up in the tree and outputs the (new) λ-term at the top (reverse construction of the first stage).

In Figure 3.3 on page 29 we included an example with $X = \widehat{X} = \{x_j \mid j \in \mathbb{N}\}$ and $L = \lambda x_0 x_3.x_1 x_0 x_2$. In this case we have $\mathrm{FV}(L) = \{x_1, x_2\}$, $n = 2$, and $\widehat{X_n} = \{\widehat{y_0} = x_0, \widehat{y_1} = x_3\}$.

We then define $\widehat{\mathrm{C}}_X(L)$ as usual. Note that this construction works for SLC and $\widehat{\mathrm{SLC}}$, therefore the two maps $\xi_X \colon \mathrm{SLC}\, X \to \widehat{\mathrm{SLC}} X$ and $\zeta_X \colon \widehat{\mathrm{SLC}} X \to \mathrm{SLC}\, X$ can be defined for all sets $X \in \mathbf{Set}^{\infty}_{\mathrm{inj}}$, all $L \in \mathrm{SLC}\, X$, and all $K \in \widehat{\mathrm{SLC}} X$ by

$$\begin{aligned} \xi_X(L) &\coloneqq ((L)_{\mathrm{C}_X})_{\widehat{\mathrm{C}}_X} = (L)_{\widehat{\mathrm{C}}_X}, \\ \zeta_X(K) &\coloneqq (K)_{\widehat{\mathrm{C}}_X}. \end{aligned}$$

As any two α-equivalent λ-terms have exactly one canonical representative with respect to either canonical form it is obvious that ξ and ζ induce a natural isomorphism of SLC_α and $\widehat{\mathrm{SLC}_\alpha}$ (as noted earlier $\mathrm{SLC}\, f$ for any morphism of sets f just renames the free variables according to f, up to α-equivalence).

Note that by introducing this canonical representative on $\widehat{\mathrm{SLC}}$ we could extend "usual" lambda calculus to the full subcategory of **Set** consisting of all infinite sets by defining $\widehat{\mathrm{SLC}} f$ for any morphism $f \colon X \to Y$ of infinite sets for all $L \in \widehat{\mathrm{SLC}} X$ as applying the first stage of the algorithm for the canonical form in $\widehat{\mathrm{SLC}} X$, then renaming all free variables according to f, and finally applying the second stage of the algorithm of the canonical form in $\widehat{\mathrm{SLC}} Y$ (by just considering the labels of the variables and ignoring that they have "wrong" names for Y). □

3.9 Canonical Form

In this section we give an informal description of an algorithm that computes the canonical form of a λ-term as described in Section 3.3.

Let $D \in \mathbf{Set}$ be given by $D \coloneqq \{d_i \mid i \in \mathbb{N}\}$ with the total order given by the indices and $X \in \mathbf{Set}$ such that $x, y, z \in X$. We just give an informal description of how such an algorithm might work as a detailed description would be tedious

and not very enlightening. It may be useful to look at the two stages of the algorithm in the example of the λ-term $(\lambda x d_1.yzd_1)(\lambda d_0 d_2.d_0 d_2(\lambda x.x))$ given in Figures 3.1 and 3.2 on pages 27 and 28.

First stage. In a first stage the algorithm decomposes the given λ-term into its components in reverse order of its building process and therefore creates a "building tree" of the λ-term. The root of the tree is given by the λ-term itself. If the λ-term is an application the algorithm just splits the λ-term into its two components. If it is an abstraction the algorithm "throws out" the outermost abstraction and labels the edge as one higher than the highest labelled edge from this node to the root (or 0 if no such edge exists) and also labels the abstracted variable(s) (if there are any) accordingly (in our example the labels are the upper indices in red). If the λ-term is a variable the algorithm stops. This process is then repeated in all subtrees until all leaves are variables.

Second stage. In the second stage the algorithm "reassembles" the λ-term with (new) variables defined as follows: Unlabelled variables stay the same (these need to be elements in X). For every edge that is labelled with a "0" the algorithm searches the highest label of any edge in the subtree starting with the source node of this edge. Let this number be $n \in \mathbb{N}$ (in our example we have $n = 1$ in the left subtree and $n = 2$ in the right subtree). Every labelled variable is in exactly one of these subtrees. If the variable is labelled with $i \in \mathbb{N}$, it is replaced by d_k with $k = n - i$ (in our example the indices of the new variables are in green). The same relabelling is also done for the labelled edges. The algorithm then goes up in the tree and outputs the (new) λ-term at the top (reverse construction of the first stage).

Note that the resulting λ-term of this process is α-equivalent to the input λ-term. Further the proposed algorithm has a running time (low-)polynomial in the degree of the λ-term.

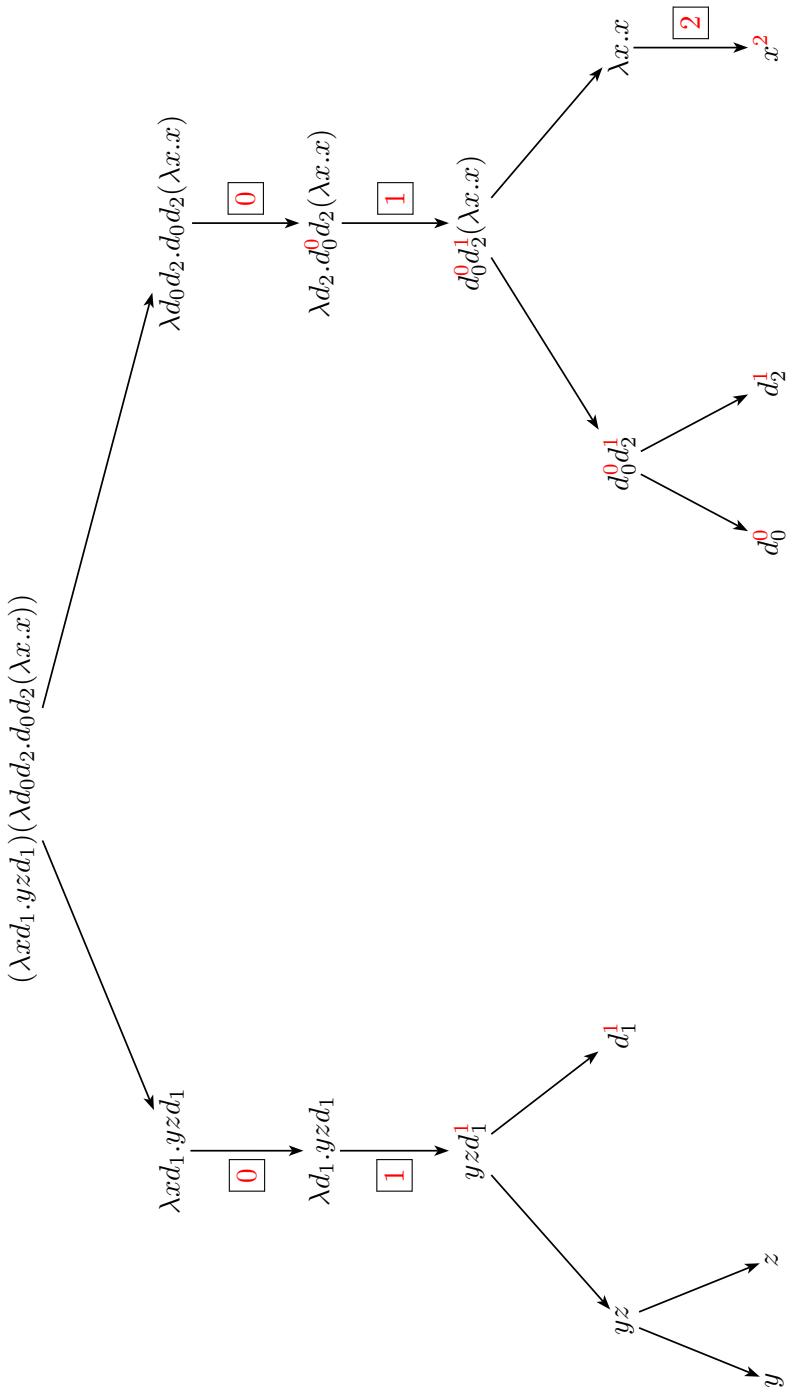


Figure 3.1: Computation tree 1.

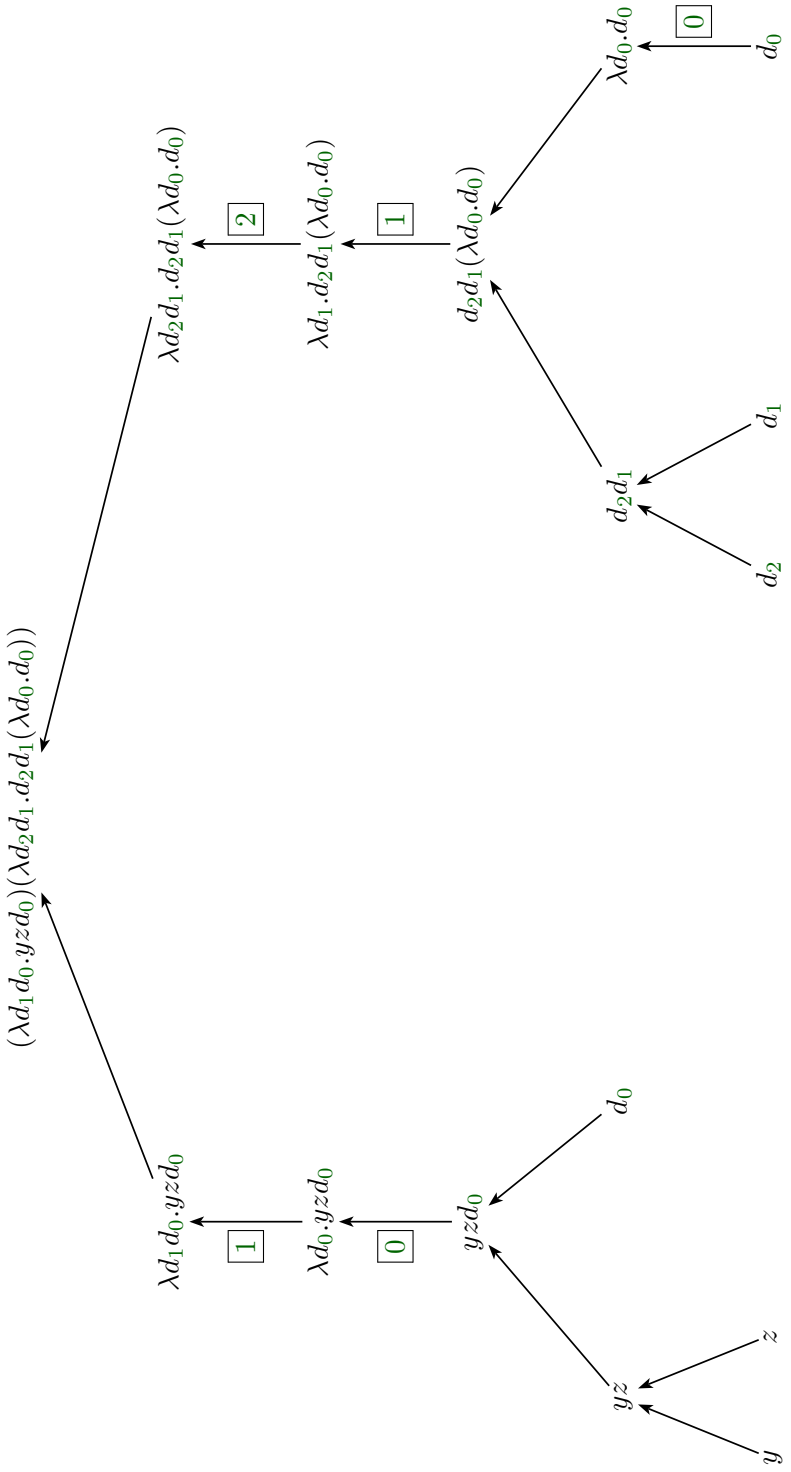


Figure 3.2: Computation tree 2.

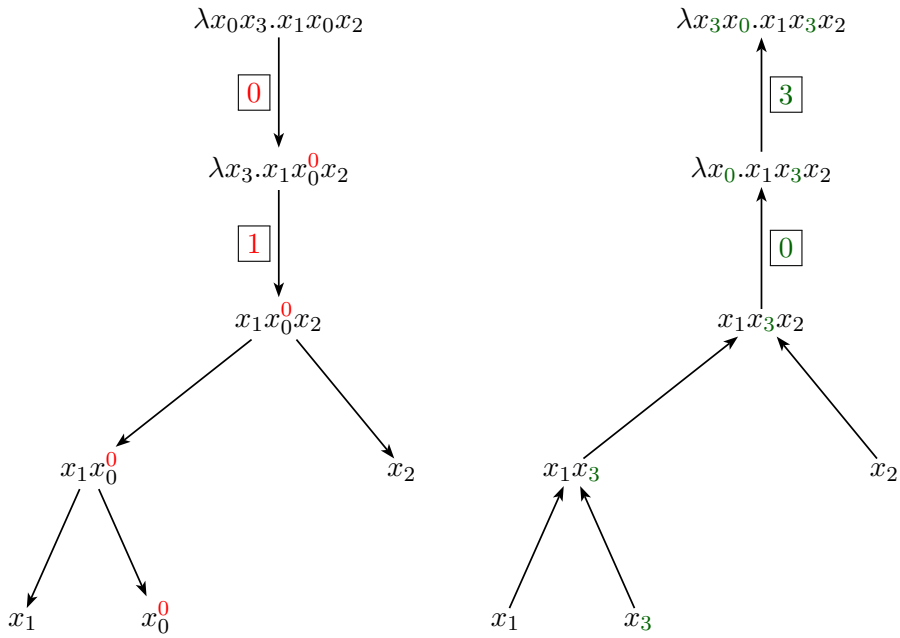


Figure 3.3: Computation trees for $\widehat{\mathrm{C}}_X(\lambda x_0 x_3.x_1 x_0 x_2)$.

Properties of Exponential Monads

In this chapter we show that two properties of lambda calculus hold in very general situations. These properties could be described as "morphisms of monads commute with substitution" and "application commutes with multiplication".

4.1 Morphisms and Substitution

For the first property consider the map $\tau\colon \mathrm{LC} \to \mathrm{LC}'$ defined for all $X \in \mathbf{Set}$ by $\tau_X := \mathrm{LC}\,\iota_X$. It is easy to see that this defines a morphism of monads. Given any $T_1, T_2 \in \mathrm{LC}\,X$ for some $X \in \mathbf{Set}$ we immediately see that

$$\tau_X(T_1)[x \leftarrow \tau_X(T_2)] = \tau_X(T_1[x \leftarrow T_2])$$

holds, i.e. it does not matter if we first apply τ_X to both terms and then substitute or if we first substitute and then apply τ_X.

This is a property that holds for all monads and morphisms between them, however we first need to define substitution for an arbitrary monad. Therefore let $(\mathrm{M}, \eta^{(\mathrm{M})}, \mu^{(\mathrm{M})})$ be any monad. Substitution in this monad is then defined as for LC in (3.9) on page 21 by replacing LC by M.

Lemma 4.1 *Let $(\mathrm{T}, \eta^{(\mathrm{T})}, \mu^{(\mathrm{T})})$ and $(\mathrm{S}, \eta^{(\mathrm{S})}, \mu^{(\mathrm{S})})$ be two monads over* **Set**. *If $\tau\colon \mathrm{T} \stackrel{\bullet}{\to} \mathrm{S}$ is a morphism of monads then it commutes with substitution in the following two ways:*

(i) For all $X \in \mathbf{Set}$, all $K_1, K_2 \in \mathrm{T}\,X$, and all $x \in X$ it holds that

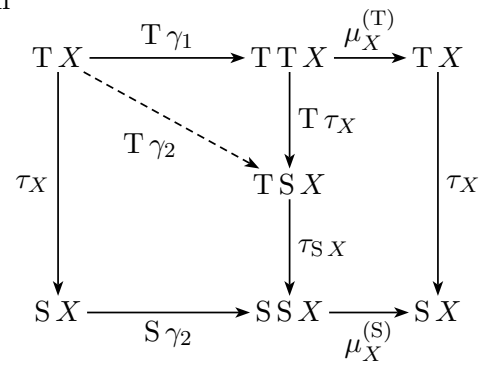
$$\tau_X(K_1)[x \leftarrow \tau_X(K_2)] = \tau_X(K_1[x \leftarrow K_2]).$$

(ii) For all $X \in \mathbf{Set}$, all $K_1 \in \mathrm{T}'\,X$, and all $K_2 \in \mathrm{T}\,X$ it holds that

$$\tau_{X^*}(K_1)[* \leftarrow \tau_X(K_2)] = \tau_X(K_1[* \leftarrow K_2]).$$

Proof For *(i)*, given the assumptions of the statement, we need to prove that

the outer rectangle in



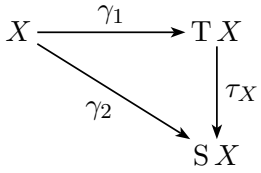
is commutative, where $\gamma_1 \colon X \to \mathrm{T}\,X$ is given for all $y \in X$ by

$$y \mapsto \begin{cases} \eta_X^{(\mathrm{T})}(y), & \text{if } x \neq y \in X, \\ K_2, & \text{if } y = x \end{cases}$$

and $\gamma_2 \colon X \to \mathrm{S}\,X$ is given for all $y \in X$ by

$$y \mapsto \begin{cases} \eta_X^{(\mathrm{S})}(y), & \text{if } x \neq y \in X, \\ \tau_X(K_2), & \text{if } y = x. \end{cases}$$

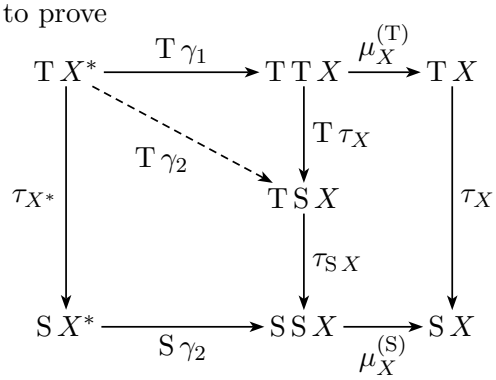
It is enough to prove the two rectangles and the triangle to be commutative. The right rectangle in this diagram commutes due to condition (2.4) in the definition of a morphism of monads. The lower left rectangle commutes as τ is natural by definition. The commutativity of the triangle follows from the commutativity of



and the fact that T is a functor. However this triangle clearly commutes: For $x \neq y \in X$ this follows from condition (2.3) in the definition of a morphism of monads and for $x \in X$ we have

$$\tau_X(\gamma_1(x)) = \tau_X(K_2) = \gamma_2(x).$$

For *(ii)* we need to prove



commutative, where $\gamma_1 \colon X^* \to \mathrm{T}\, X$ is given for all $y \in X^*$ by

$$y \mapsto \begin{cases} \eta_X^{(\mathrm{T})}(y), & \text{if } y \in X, \\ K_2, & \text{if } y = *\end{cases}$$

and $\gamma_2 \colon X^* \to \mathrm{S}\, X$ is given for all $y \in X^*$ by

$$y \mapsto \begin{cases} \eta_X^{(\mathrm{S})}(y), & \text{if } y \in X, \\ \tau_X(K_2), & \text{if } y = *. \end{cases}$$

This follows by the same argument as in the previous case. □

4.2 Application and Multiplication

For the second property consider the monad LC again. We have that application and multiplication commute, i.e. for all $X \in \mathbf{Set}$ and all $T_1, T_2 \in \mathrm{LC}\,\mathrm{LC}\, X$ the equation

$$\mu_X^{(\mathrm{LC})}(\mathrm{app}_{\mathrm{LC}\, X}(T_1, T_2)) = \mathrm{app}_X(\mu_X^{(\mathrm{LC})}(T_1), \mu_X^{(\mathrm{LC})}(T_2))$$

holds.

To generalize this statement we need to define application for general monads. However we only do this for monads together with a morphism of modules from the tautological module to its derived module. As for substitution we describe application in categorical terms such that this definition coincides with the usual one in the case of LC (and SLC).

For any monad M with a morphism of modules $\varepsilon^{(\mathrm{M})} \colon \mathrm{M} \dot{\to} \mathrm{M}'$ we define for all $X, Y \in \mathbf{Set}$ a map $\mathrm{app}_{X,Y}^{(\mathrm{M})} \colon \mathrm{M}\, X \times \mathrm{M}\, Y \to \mathrm{M}(X \amalg Y)$ by the following chain of maps:

$$\begin{aligned} \mathrm{M}\, X \times \mathrm{M}\, Y &\xrightarrow{\varepsilon_X^{(\mathrm{M})} \times \mathrm{Id}_{\mathrm{M}\, Y}} \mathrm{M}\, X^* \times \mathrm{M}\, Y \xrightarrow{k_{X,Y}^{(\mathrm{M})}} \mathrm{M}(X \amalg \mathrm{M}\, Y) \xrightarrow{l_{X,Y}^{(\mathrm{M})}} \cdots \\ &\cdots \xrightarrow{l_{X,Y}^{(\mathrm{M})}} \mathrm{M}\,\mathrm{M}(X \amalg Y) \xrightarrow{\mu_{X \amalg Y}^{(\mathrm{M})}} \mathrm{M}(X \amalg Y), \end{aligned}$$

where $k_{X,Y}^{(\mathrm{M})} \colon \mathrm{M}\, X^* \times \mathrm{M}\, Y \to \mathrm{M}(X \amalg \mathrm{M}\, Y)$ is determined by

$$\begin{aligned} r_{X,Y}^{(\mathrm{M})} \colon \mathrm{M}\, Y &\to \mathrm{Hom}(X^*, X \amalg \mathrm{M}\, Y) \\ T &\mapsto \begin{cases} x \mapsto x, & \text{if } x \in X, \\ * \mapsto T \end{cases} \end{aligned}$$

and $l_{X,Y}^{(\mathrm{M})} \colon \mathrm{M}(X \amalg \mathrm{M}\, Y) \to \mathrm{M}\,\mathrm{M}(X \amalg Y)$ is induced by

$$\begin{aligned} s_{X,Y}^{(\mathrm{M})} \colon X \amalg \mathrm{M}\, Y &\to \mathrm{M}(X \amalg Y), \\ z &\mapsto \begin{cases} \eta_{X \amalg Y}^{(\mathrm{M})}(z), & \text{if } z \in X, \\ \mathrm{M}\, j_Y(z), & \text{if } z \in \mathrm{M}\, Y. \end{cases} \end{aligned}$$

For $X = Y$ we can further define a map $\mathrm{app}^{(\mathrm{M})}_X : \mathrm{M}\,X \times \mathrm{M}\,X \to \mathrm{M}\,X$ by the composition of $\mathrm{app}^{(\mathrm{M})}_{X,X}$ with the map induced by the map ∇_X from $X \amalg X$ to X which sends every element in $X \amalg X$ to the one of the same name in X (we give a shorter description of this map after Lemma 4.4).

Note that in the cases of SLC and SLC_δ, for $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$, with $\varepsilon = \mathrm{app1}$ we have that $\mathrm{app}_X = \mathrm{app}^{(\mathrm{SLC}_\delta)}_X$ holds by definition, i.e. we do not introduce any ambiguity by using this notation. Further it is not hard to see that $\mathrm{app}^{(\mathrm{M})}_-$ already determines $\mathrm{app}^{(\mathrm{M})}_{-,-}$ by

$$\mathrm{M}\,X \times \mathrm{M}\,Y \xrightarrow{\mathrm{M}\,j_X \times \mathrm{M}\,j_Y} \mathrm{M}(X \amalg Y) \times \mathrm{M}(X \amalg Y) \xrightarrow{\mathrm{app}^{(\mathrm{M})}_{X \amalg Y}} \mathrm{M}(X \amalg Y).$$

It is also important to note that $\mathrm{app}^{(\mathrm{M})}_{-,-}$ and $\mathrm{app}^{(\mathrm{M})}_-$ form natural transformations as we now prove.

Lemma 4.2 *With the notation above the map* $\mathrm{app}^{(\mathrm{M})}_{-,-} : \mathrm{M} \times \mathrm{M} \to \mathrm{G}$ *is a natural transformation of bifunctors, where* $\mathrm{G} : \mathbf{Set} \times \mathbf{Set} \to \mathbf{Set}$ *is given by the composition of the functor that takes two sets to their disjoint union with* M.

Proof Let $f : X \to A$ and $g : Y \to B$ be any morphisms in **Set**. We need to prove the outside of

$$\begin{array}{ccc}
\mathrm{M}\,X \times \mathrm{M}\,Y & \xrightarrow{\mathrm{M}\,f \times \mathrm{M}\,g} & \mathrm{M}\,A \times \mathrm{M}\,B \\
\downarrow{\scriptstyle \varepsilon^{(\mathrm{M})}_X \times \mathrm{Id}_{\mathrm{M}\,Y}} & & \downarrow{\scriptstyle \varepsilon^{(\mathrm{M})}_A \times \mathrm{Id}_{\mathrm{M}\,B}} \\
\mathrm{M}\,X^* \times \mathrm{M}\,Y & \xrightarrow{\mathrm{M}\,f^* \times \mathrm{M}\,g} & \mathrm{M}\,A^* \times \mathrm{M}\,B \\
\downarrow{\scriptstyle k^{(\mathrm{M})}_{\mathrm{M}\,X,\mathrm{M}\,Y}} & & \downarrow{\scriptstyle k^{(\mathrm{M})}_{\mathrm{M}\,A,\mathrm{M}\,B}} \\
\mathrm{M}(X \amalg \mathrm{M}\,Y) & \xrightarrow{\mathrm{M}(f \amalg \mathrm{M}\,g)} & \mathrm{M}(A \amalg \mathrm{M}\,B) \\
\downarrow{\scriptstyle l^{(\mathrm{M})}_{\mathrm{M}\,X,\mathrm{M}\,Y}} & & \downarrow{\scriptstyle l^{(\mathrm{M})}_{\mathrm{M}\,A,\mathrm{M}\,B}} \\
\mathrm{M}\,\mathrm{M}(X \amalg Y) & \xrightarrow{\mathrm{M}\,\mathrm{M}(f \amalg g)} & \mathrm{M}\,\mathrm{M}(A \amalg B) \\
\downarrow{\scriptstyle \mu^{(\mathrm{M})}_{X \amalg Y}} & & \downarrow{\scriptstyle \mu^{(\mathrm{M})}_{A \amalg B}} \\
\mathrm{M}(X \amalg Y) & \xrightarrow{\mathrm{M}(f \amalg g)} & \mathrm{M}(A \amalg B)
\end{array}$$

commutative.

The top and bottom rectangles in this diagram commute as $\varepsilon^{(\mathrm{M})}$, Id, and $\mu^{(\mathrm{M})}$ are natural transformations. For the two rectangles in the middle it is

enough to prove that the maps that induce $k^{(\mathrm{M})}_{-,-}$ and $l^{(\mathrm{M})}_{-,-}$ satisfy the corresponding properties.

For $k^{(\mathrm{M})}_{-,-}$ this means proving

$$\begin{array}{ccc} X^* \times \mathrm{M}\,Y & \xrightarrow{f^* \times \mathrm{M}\,g} & A^* \times \mathrm{M}\,B \\ \downarrow & & \downarrow \\ X \amalg \mathrm{M}\,Y & \xrightarrow[f \amalg \mathrm{M}\,g]{} & A \amalg \mathrm{M}\,B \end{array}$$

commutative, where the vertical maps are the inducing ones. By first going down and then right we get

$$(y, T) \mapsto \begin{cases} (f \amalg \mathrm{M}\,g)(y), & \text{if } y \in X, \\ (f \amalg \mathrm{M}\,g)(T), & \text{if } y = *\end{cases}$$

for all $y \in X^*$ and all $T \in \mathrm{M}\,Y$. By first going right and then down we get

$$(y, T) \mapsto (f^*(y), \mathrm{M}\,g(T)) \mapsto \begin{cases} f^*(y), & \text{if } f^*(y) \in A, \\ \mathrm{M}\,g(T), & \text{if } f^*(y) = *\end{cases}$$

for all $y \in X^*$ and all $T \in \mathrm{M}\,Y$. As we have

$$f^*(y) = * \quad \Leftrightarrow \quad y = *$$

and $f^*|_X = \iota_A \circ f$, we can rewrite the second composite as

$$(y, T) \mapsto \begin{cases} f(y), & \text{if } y \in X, \\ \mathrm{M}\,g(T), & \text{if } y = *. \end{cases}$$

That the two composites agree follows by the definition of the disjoint union.

For $l^{(\mathrm{M})}_{-,-}$ this means proving

$$\begin{array}{ccc} X \amalg \mathrm{M}\,Y & \xrightarrow{f \amalg \mathrm{M}\,g} & A \amalg \mathrm{M}\,B \\ \downarrow{\scriptstyle s^{(\mathrm{M})}_{X,Y}} & & \downarrow{\scriptstyle s^{(\mathrm{M})}_{A,B}} \\ \mathrm{M}(X \amalg Y) & \xrightarrow[\mathrm{M}(f \amalg g)]{} & \mathrm{M}(A \amalg B) \end{array}$$

commutative. By definition we have

$$\mathrm{M}(f \amalg g) \circ s^{(\mathrm{M})}_{X,Y} = \begin{cases} x \mapsto \eta^{(\mathrm{M})}_{X \amalg Y}(x) \mapsto \mathrm{M}(f \amalg g)(\eta^{(\mathrm{M})}_{X \amalg Y}(x)), & \text{if } x \in X, \\ L \mapsto \mathrm{M}\,j_Y(L) \mapsto \mathrm{M}(f \amalg g)(\mathrm{M}\,j_Y(L)), & \text{if } L \in \mathrm{M}\,Y \end{cases}$$

and

$$s^{(\mathrm{M})}_{A,B} \circ f \amalg \mathrm{M}\,g = \begin{cases} x \mapsto (f \amalg \mathrm{M}\,g)(x) \mapsto \eta^{(\mathrm{M})}_{A \amalg B}((f \amalg \mathrm{M}\,g)(x)), & \text{if } x \in X, \\ L \mapsto (f \amalg \mathrm{M}\,g)(L) \mapsto \mathrm{M}\,j_B((f \amalg \mathrm{M}\,g)(L)), & \text{if } L \in \mathrm{M}\,Y. \end{cases}$$

We can now conclude as

$$\mathrm{M}(f \amalg g)(\eta^{(\mathrm{M})}_{X \amalg Y}(x)) = \eta^{(\mathrm{M})}_{A \amalg B}((f \amalg \mathrm{M}\, g)(x))$$

holds for all $x \in X$ by the naturality of $\eta^{(\mathrm{M})}$ and

$$\mathrm{M}(f \amalg g)(\mathrm{M}\, j_Y(L)) = \mathrm{M}\, j_B((f \amalg \mathrm{M}\, g)(L))$$

holds for all $L \in \mathrm{M}\, Y$ as M is a functor and as $j_B \circ g = (f \amalg g) \circ j_Y$ holds. □

Corollary 4.3 $\mathrm{app}^{(\mathrm{M})}_{-} \colon (\mathrm{M} \times \mathrm{M}) \circ \Delta \to \mathrm{M}$ *is a natural transformation, where* Δ *denotes the diagonal functor.*

Proof Let $f \colon X \to A$ be any morphism in **Set**. With Lemma 4.2 at hand it is only left to show that

$$\begin{array}{ccc} \mathrm{M}(X \amalg X) & \xrightarrow{\mathrm{M}(f \amalg f)} & \mathrm{M}(A \amalg A) \\ \downarrow{\scriptstyle \mathrm{M}\, \nabla_X} & & \downarrow{\scriptstyle \mathrm{M}\, \nabla_A} \\ \mathrm{M}\, X & \xrightarrow[\mathrm{M}\, f]{} & \mathrm{M}\, A \end{array}$$

commutes. This follows from the commutativity of the square

$$\begin{array}{ccc} X \amalg X & \xrightarrow{f \amalg f} & A \amalg A \\ \downarrow{\scriptstyle \nabla_X} & & \downarrow{\scriptstyle \nabla_A} \\ X & \xrightarrow[f]{} & A \end{array}$$

and functoriality. □

Before we give the actual statement we need a preliminary result that will allow us to use an alternative description of $\mathrm{app}^{(\mathrm{M})}_{-}$.

Lemma 4.4 *With the notation above the diagram*

$$\begin{array}{ccccc} \mathrm{M}\, X^* \times \mathrm{M}\, X & \xrightarrow{k^{(\mathrm{M})}_{X,X}} & \mathrm{M}(X \amalg \mathrm{M}\, X) & \xrightarrow{l^{(\mathrm{M})}_{X,X}} & \mathrm{M}\, \mathrm{M}(X \amalg X) \\ \downarrow{\scriptstyle m^{(\mathrm{M})}_X} & & & & \downarrow{\scriptstyle \mu^{(\mathrm{M})}_{X \amalg X}} \\ \mathrm{M}\, \mathrm{M}\, X & \xrightarrow[\mu^{(\mathrm{M})}_X]{} & \mathrm{M}\, X & \xleftarrow[\mathrm{M}\, \nabla_X]{} & \mathrm{M}(X \amalg X) \end{array}$$

commutes for all $X \in \mathbf{Set}$*, where* $m_X^{(\mathrm{M})} \colon \mathrm{M}\, X^* \times \mathrm{M}\, X \to \mathrm{M}\,\mathrm{M}\, X$ *is defined as the map induced by*

$$t_X^{(\mathrm{M})} \colon \mathrm{M}\, X \to \mathrm{Hom}(X^*, \mathrm{M}\, X)$$
$$T \mapsto \begin{cases} x \mapsto \eta_X^{(\mathrm{M})}(x), & \text{if } x \in X, \\ * \mapsto T. \end{cases}$$

Proof Let $X \in \mathbf{Set}$ be arbitrary. Note that

$$\mathrm{M}\, \nabla_X \circ \mu_{X \amalg X}^{(\mathrm{M})} = \mu_X^{(\mathrm{M})} \circ \mathrm{M}\,\mathrm{M}\, \nabla_X$$

holds by the naturality of $\mu^{(\mathrm{M})}$. It is therefore sufficient to prove

$$\begin{array}{ccccc} \mathrm{M}\, X^* \times \mathrm{M}\, X & \xrightarrow{k_{X,X}^{(\mathrm{M})}} & \mathrm{M}(X \amalg \mathrm{M}\, X) & \xrightarrow{l_{X,X}^{(\mathrm{M})}} & \mathrm{M}\,\mathrm{M}(X \amalg X) \\ \downarrow{\scriptstyle m_X^{(\mathrm{M})}} & & & & \downarrow{\scriptstyle \mathrm{M}\,\mathrm{M}\, \nabla_X} \\ \mathrm{M}\,\mathrm{M}\, X & \xrightarrow[\mu_X^{(\mathrm{M})}]{} & \mathrm{M}\, X & \xleftarrow[\mu_X^{(\mathrm{M})}]{} & \mathrm{M}\,\mathrm{M}\, X \end{array}$$

commutative. By adding the identities between the two $\mathrm{M}\,\mathrm{M}\, X$ sets it follows that proving

$$\begin{array}{ccc} \mathrm{M}\, X^* \times \mathrm{M}\, X & \xrightarrow{k_{X,X}^{(\mathrm{M})}} & \mathrm{M}(X \amalg \mathrm{M}\, X) \\ \downarrow{\scriptstyle m_X^{(\mathrm{M})}} & & \downarrow{\scriptstyle l_{X,X}^{(\mathrm{M})}} \\ \mathrm{M}\,\mathrm{M}\, X & \xleftarrow[\mathrm{M}\,\mathrm{M}\, \nabla_X]{} & \mathrm{M}\,\mathrm{M}(X \amalg X) \end{array}$$

commutative already implies the result.

That this holds can be seen by considering the maps that induce the ones above, which is shown in

$$\begin{array}{ccc} \mathrm{M}\, X & \xrightarrow{r_{X,X}^{(\mathrm{M})}} & \mathrm{Hom}(X^*, X \amalg \mathrm{M}\, X) \\ \downarrow{\scriptstyle t_X^{(\mathrm{M})}} & & \downarrow{\scriptstyle s_{X,X_*}^{(\mathrm{M})}} \\ \mathrm{Hom}(X^*, \mathrm{M}\, X) & \xleftarrow[(\mathrm{M}\, \nabla_X)_*]{} & \mathrm{Hom}(X^*, \mathrm{M}(X \amalg X)), \end{array}$$

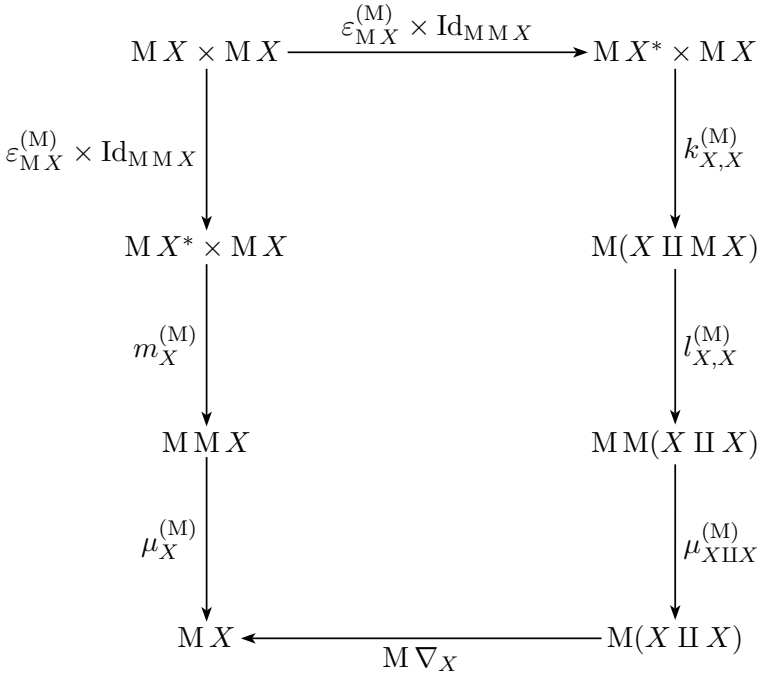
where a lower star denotes postcomposition by that map.

Plugging in the definitions we see that $(\mathrm{M}\, \nabla_X)_* \circ s_{X,X_*}^{(\mathrm{M})} \circ r_{X,X}^{(\mathrm{M})}$ is given for all $T \in \mathrm{M}\, X$ by

$$T \mapsto \begin{cases} x \mapsto x \mapsto \eta_X^{(\mathrm{M})}(x) \mapsto \eta_X^{(\mathrm{M})}(x), & \text{if } x \in X, \\ * \mapsto T \mapsto \mathrm{M}\, j_X(T) \mapsto T, \end{cases}$$

where we used $\nabla_X \circ j_X = \mathrm{Id}_X$ for the last step. However this is exactly the definition of $t_X^{(\mathrm{M})}$ and hence finishes the proof. □

This lemma implies that



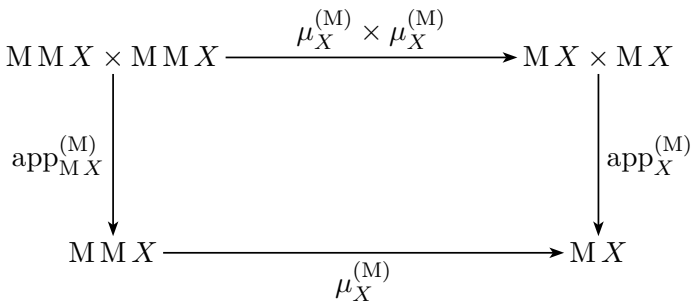
commutes for all $X \in \mathbf{Set}$, i.e. it gives an alternative way of defining $\mathrm{app}^{(\mathrm{M})}_-$. This notation is going to make the presentation of the next proof simpler.

Lemma 4.5 *Let M and $\varepsilon^{(\mathrm{M})}$ be as above. Then the identity*

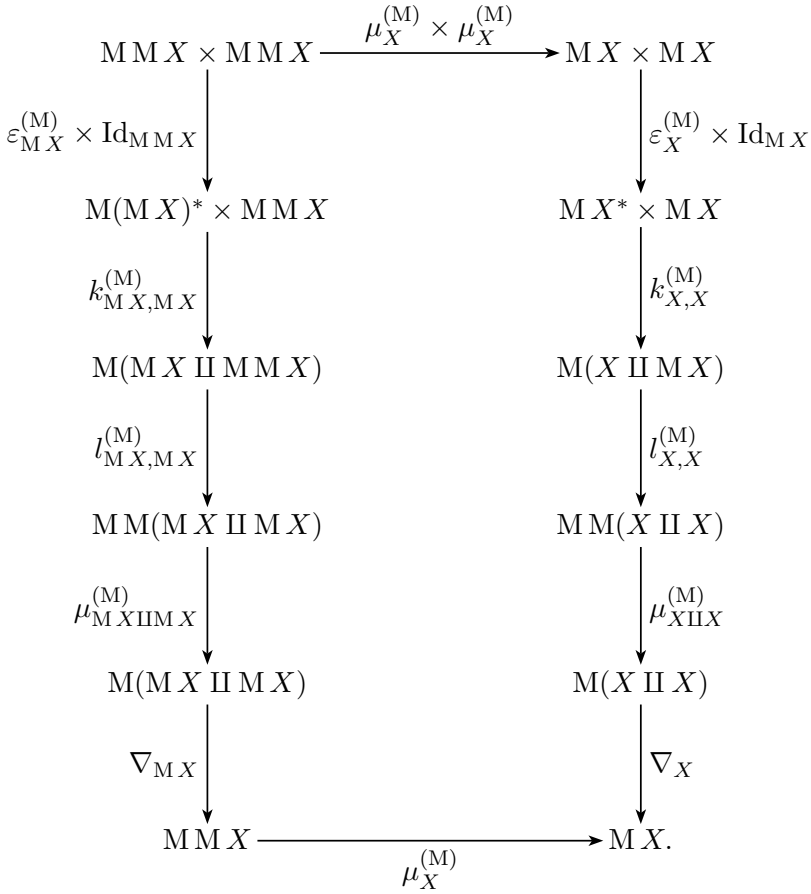
$$\mu^{(\mathrm{M})}_X(\mathrm{app}^{(\mathrm{M})}_{\mathrm{M}\,X}(L_1, L_2)) = \mathrm{app}^{(\mathrm{M})}_X(\mu^{(\mathrm{M})}_X(L_1), \mu^{(\mathrm{M})}_X(L_2))$$

holds for all $X \in \mathbf{Set}$ and all $L_1, L_2 \in \mathrm{M}\,\mathrm{M}\,X$.

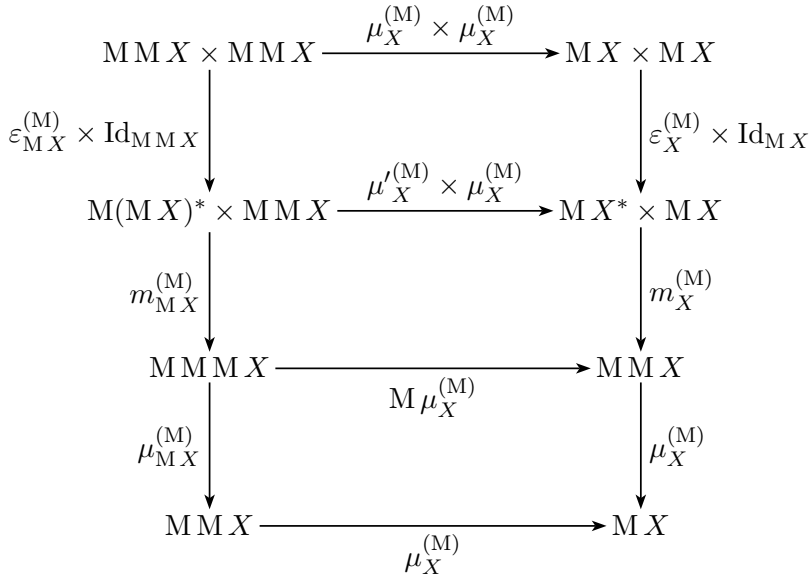
Proof Let $X \in \mathbf{Set}$ be arbitrary. We need to prove



commutative. Plugging in the definitions shows that this diagram is



Using Lemma 4.4 this is equivalent to proving the outside of



(4.1)

commutative. In this diagram the upper rectangle commutes by the componentwise definition of the product of two modules and the assumption that $\varepsilon^{(\mathrm{M})}$ is a morphism of modules. Further the lower rectangle commutes by the associativity of $\mu^{(\mathrm{M})}$. However the middle triangle is in general *not* commutative, e.g. for LC it will not commute. Our goal is to prove that the composition of the lower two rectangles is commutative.

By fixing $T \in \mathrm{M}\,\mathrm{M}\,X$ the middle rectangle becomes

$$\begin{array}{ccc} \mathrm{M}(\mathrm{M}\,X)^* & \xrightarrow{\mu'^{(\mathrm{M})}_X} & \mathrm{M}\,X^* \\ \Big\downarrow{\scriptstyle m^{(\mathrm{M})}_{\mathrm{M}\,X,T}} & & \Big\downarrow{\scriptstyle m^{(\mathrm{M})}_{X,\mu^{(\mathrm{M})}(T)}} \\ \mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X & \xrightarrow[\mathrm{M}\,\mu^{(\mathrm{M})}_X]{} & \mathrm{M}\,\mathrm{M}\,X, \end{array}$$

where $m^{(\mathrm{M})}_{\mathrm{M}\,X,T} \colon \mathrm{M}(\mathrm{M}\,X)^* \to \mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X$ is induced by

$$\begin{aligned} u^{(\mathrm{M})}_{\mathrm{M}\,X,T} &\colon (\mathrm{M}\,X)^* \to \mathrm{M}\,\mathrm{M}\,X \\ L &\mapsto \begin{cases} \eta^{(\mathrm{M})}_{\mathrm{M}\,X}(L), & \text{if } L \in \mathrm{M}\,X, \\ T, & \text{if } L = *\end{cases} \end{aligned}$$

and $m^{(\mathrm{M})}_{X,\mu^{(\mathrm{M})}(T)} \colon \mathrm{M}\,X^* \to \mathrm{M}\,\mathrm{M}\,X$ is induced by

$$\begin{aligned} v^{(\mathrm{M})}_{X,\mu^{(\mathrm{M})}(T)} &\colon X^* \to \mathrm{M}\,X \\ y &\mapsto \begin{cases} \eta^{(\mathrm{M})}_X(y), & \text{if } y \in X, \\ \mu^{(\mathrm{M})}(T), & \text{if } y = *. \end{cases} \end{aligned}$$

Recall that the map $g_X : (\mathrm{M}\,X)^* \to \mathrm{M}\,X^*$ is defined by $g|_{\mathrm{M}\,X} = \mathrm{M}\,\iota_X$ and $g(*) = \eta^{(\mathrm{M})}_{X^*}(*)$. This induces the map $\mathrm{M}\,g_X$ from $\mathrm{M}(\mathrm{M}\,X)^*$ to $\mathrm{M}\,\mathrm{M}\,X^*$. By Lemma 4.6 below the diagram

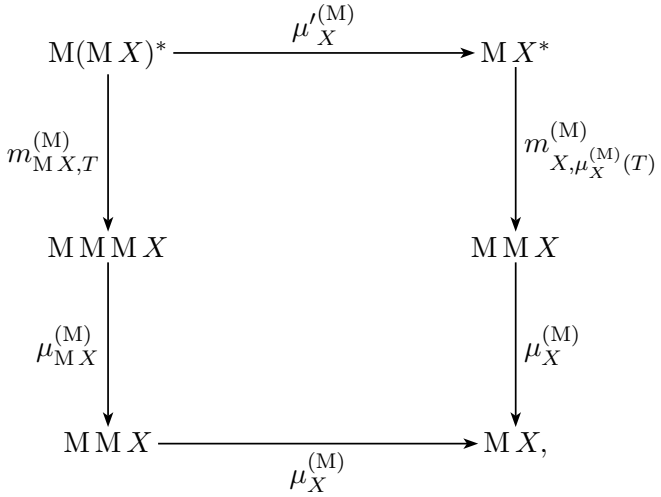
$$\begin{array}{ccc} \mathrm{M}\,\mathrm{M}\,X^* & \xrightarrow{\mu^{(\mathrm{M})}_{X^*}} & \mathrm{M}\,X^* \\ \Big\downarrow{\scriptstyle \tilde{m}^{(\mathrm{M})}_{X,T}} & & \Big\downarrow{\scriptstyle m^{(\mathrm{M})}_{X,\mu^{(\mathrm{M})}(T)}} \\ \mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X & \xrightarrow[\mu^{(\mathrm{M})}_{\mathrm{M}\,X}]{} & \mathrm{M}\,\mathrm{M}\,X \end{array} \tag{4.2}$$

commutes for all $X \in \mathbf{Set}$, where $\tilde{m}^{(\mathrm{M})}_{X,T} \colon \mathrm{M}\,\mathrm{M}\,X^* \to \mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X$ is induced by

$$\begin{aligned} w^{(\mathrm{M})}_{X,T} &\colon X^* \to \mathrm{M}\,\mathrm{M}\,X \\ y &\mapsto \begin{cases} \eta^{(\mathrm{M})}_{\mathrm{M}\,X}(\eta^{(\mathrm{M})}_X(y)), & \text{if } y \in X, \\ T, & \text{if } y = *\end{cases} \end{aligned}$$

(first it induces a map from $\mathrm{M}\,\mathrm{M}\,X^*$ to $\mathrm{M}\,\mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X$ and we then compose with $\mathrm{M}\,\mathrm{M}\,\mu_X^{(\mathrm{M})}$).

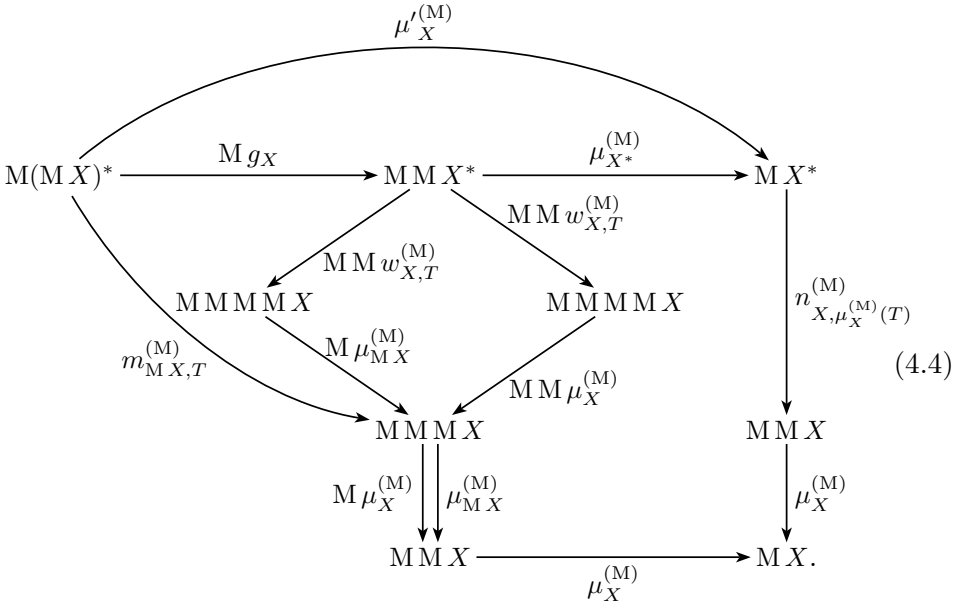
We already know that the first rectangle in (4.1) commutes, hence it is enough to prove the lower part commutative. If we fix $T \in \mathrm{M}\,\mathrm{M}\,X$ we get



(4.3)

where the second index is a reference to the fixed element.

The commutativity of this diagram follows from



(4.4)

In this diagram the top commutes by the definition of $\mu'^{(\mathrm{M})}_X$. Further the right figure commutes due to Lemma 4.6. Note that the diamond in the middle does *not* commute in general. However if we go from $\mathrm{M}\,\mathrm{M}\,X^*$ all the way down to $\mathrm{M}\,X$ it does not matter which side of the diamond we take by the associativity condition of a monad.

It is left to show that going from $\mathrm{M}(\mathrm{M}\,X)^*$ to $\mathrm{M}\,\mathrm{M}\,X$ via $m^{(\mathrm{M})}_{\mathrm{M}\,X,T}$ is the same

as using the left side of the diamond, i.e. we want to prove

$$\begin{array}{ccc}
\mathrm{M}(\mathrm{M}\,X)^* & \xrightarrow{\mathrm{M}\,g_X} & \mathrm{M}\,\mathrm{M}\,X^* \\
\Big\downarrow{\scriptstyle m^{(\mathrm{M})}_{\mathrm{M}\,X,T}} & & \Big\downarrow{\scriptstyle \mathrm{M}\,\mathrm{M}\,w^{(\mathrm{M})}_{X,T}} \\
\mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X & & \mathrm{M}\,\mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X \\
\Big\downarrow{\scriptstyle \mathrm{M}\,\mu^{(\mathrm{M})}_X} & & \Big\downarrow{\scriptstyle \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X}} \\
\mathrm{M}\,\mathrm{M}\,X & \xleftarrow[\mathrm{M}\,\mu^{(\mathrm{M})}_X]{} & \mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X
\end{array}$$

commutative.

First we consider $* \in (\mathrm{M}\,X)^*$ as we have there that the rectangle in (4.4) already commutes (equivalently we could draw identity arrows between the two $\mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X$ sets). Further we use functoriality to make our lives easier, i.e. it suffices to prove

$$\begin{array}{ccc}
(\mathrm{M}\,X)^* & \xrightarrow{g_X} & \mathrm{M}\,X^* \\
\Big\downarrow{\scriptstyle u^{(\mathrm{M})}_{\mathrm{M}\,X,T}} & & \Big\downarrow{\scriptstyle \mathrm{M}\,w^{(\mathrm{M})}_{X,T}} \\
\mathrm{M}\,\mathrm{M}\,X & \xleftarrow[\mu^{(\mathrm{M})}_{\mathrm{M}\,X}]{} & \mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X
\end{array}$$

commutative. By using the naturality of $\eta^{(\mathrm{M})}$ we see that

$$\begin{array}{ccc}
X^* & \xrightarrow{\eta^{(\mathrm{M})}_{X^*}} & \mathrm{M}\,X^* \\
\Big\downarrow{\scriptstyle w^{(\mathrm{M})}_{X,T}} & & \Big\downarrow{\scriptstyle \mathrm{M}\,w^{(\mathrm{M})}_{X,T}} \\
\mathrm{M}\,\mathrm{M}\,X & \xrightarrow[\eta^{(\mathrm{M})}_{\mathrm{M}\,\mathrm{M}\,X}]{} & \mathrm{M}\,\mathrm{M}\,\mathrm{M}\,X
\end{array}$$

commutes. Hence we have

$$\begin{aligned}
\mu_{\mathrm{M}\,X}(\mathrm{M}\,w^{(\mathrm{M})}_{X,T}(g_X(*))) &= \mu_{\mathrm{M}\,X}(\mathrm{M}\,w^{(\mathrm{M})}_{X,T}(\eta_{X^*}(*))) \\
&= \mu_{\mathrm{M}\,X}(\eta^{(\mathrm{M})}_{\mathrm{M}\,\mathrm{M}\,X}(w^{(\mathrm{M})}_{X,T}(*))) \\
&= w^{(\mathrm{M})}_{X,T}(*) \\
&= T \\
&= u^{(\mathrm{M})}_{\mathrm{M}\,X,T}(*),
\end{aligned}$$

where we use the definitions of the maps involved and for the third equation the property of the unit of a monad.

Now let $L \in \mathrm{M}\,X$ be arbitrary. Note that we have $w^{(\mathrm{M})}_{X,T}\big|_X = \eta^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \eta^{(\mathrm{M})}_X$. As we clearly have $w^{(\mathrm{M})}_{X,T} \circ \iota_X = w^{(\mathrm{M})}_{X,T}\big|_X$ we get (using functoriality, the definitions of the various maps, and the properties of the unit of a monad):

$$\begin{aligned}
&\mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}\,w^{(\mathrm{M})}_{X,T} \circ \mathrm{M}\,g_X \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}\,w^{(\mathrm{M})}_{X,T} \circ \mathrm{M}(g_X(L)) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}\,w^{(\mathrm{M})}_{X,T} \circ \mathrm{M}(\mathrm{M}\,\iota_X(L)) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}\,w^{(\mathrm{M})}_{X,T} \circ \mathrm{M}\,\mathrm{M}\,\iota_X \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}(w^{(\mathrm{M})}_{X,T} \circ \iota_X) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}(w^{(\mathrm{M})}_{X,T}\big|_X) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}(\eta^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \eta^{(\mathrm{M})}_X) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}(\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\eta^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\eta^{(\mathrm{M})}_X) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}(\mathrm{M}\,\eta^{(\mathrm{M})}_X) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}(\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\eta^{(\mathrm{M})}_X) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}(\mathrm{Id}_{\mathrm{M}\,X}) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{Id}_{\mathrm{M}\,\mathrm{M}\,X} \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}(L) \\
&\qquad = \mathrm{M}(\mathrm{Id}_{\mathrm{M}\,X}) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}(\mu^{(\mathrm{M})}_X \circ \eta^{(\mathrm{M})}_{\mathrm{M}\,X}) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}(\eta^{(\mathrm{M})}_{\mathrm{M}\,X}) \circ \mathrm{M}(L) \\
&\qquad = \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ m^{(\mathrm{M})}_{\mathrm{M}\,X,T}(L).
\end{aligned}$$

We can now conclude the proof by showing (4.3) commutative. All transformations below can be visualized in (4.4). By the associativity of a monad we have

$$\mu^{(\mathrm{M})}_X \circ \mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ m^{(\mathrm{M})}_{\mathrm{M}\,X,T} = \mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ m^{(\mathrm{M})}_{\mathrm{M}\,X,T}.$$

The argument just above implies

$$\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ m^{(\mathrm{M})}_{\mathrm{M}\,X,T} = \mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}\,w^{(\mathrm{M})}_{X,T} \circ \mathrm{M}\,g_X.$$

Again by the associativity of a monad it follows that

$$\begin{aligned}
&\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}\,w^{(\mathrm{M})}_{X,T} \circ \mathrm{M}\,g_X \\
&\qquad = \mu^{(\mathrm{M})}_X \circ \mu^{(\mathrm{M})}_{\mathrm{M}\,X} \circ \mathrm{M}\,\mathrm{M}\,\mu^{(\mathrm{M})}_X \circ \mathrm{M}\,\mathrm{M}\,w^{(\mathrm{M})}_{X,T} \circ \mathrm{M}\,g_X
\end{aligned}$$

holds. By Lemma 4.6 we then have

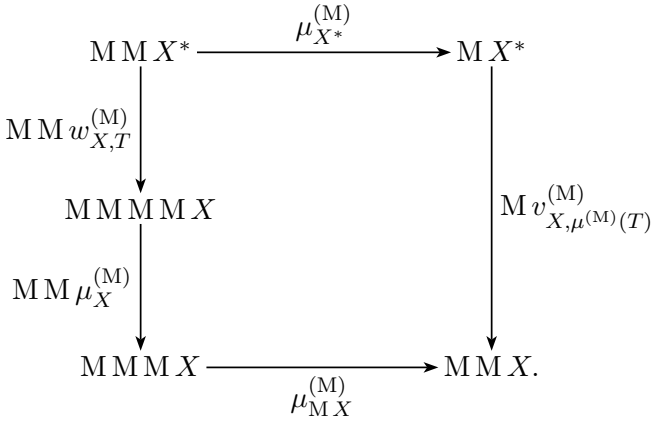
$$\begin{aligned}\mu_X^{(\mathrm{M})} \circ \mu_{\mathrm{M}\,X}^{(\mathrm{M})} \circ \mathrm{M}\,\mathrm{M}\,\mu_X^{(\mathrm{M})} \circ \mathrm{M}\,\mathrm{M}\,w_{X,T}^{(\mathrm{M})} \circ \mathrm{M}\,g_X \\ = \mu_X^{(\mathrm{M})} \circ m_{X,\mu_X^{(\mathrm{M})}(T)}^{(\mathrm{M})} \circ \mu_{X^*}^{(\mathrm{M})} \circ \mathrm{M}\,g_X\end{aligned}$$

Finally we can conclude by using the definition of ${\mu'}_X^{(\mathrm{M})}$ to get

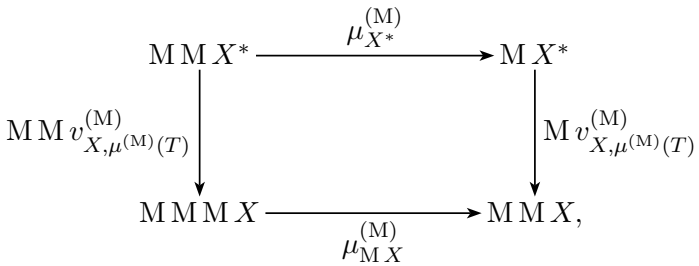
$$\mu_X^{(\mathrm{M})} \circ m_{X,\mu_X^{(\mathrm{M})}(T)}^{(\mathrm{M})} \circ \mu_{X^*}^{(\mathrm{M})} \circ \mathrm{M}\,g_X = \mu_X^{(\mathrm{M})} \circ m_{X,\mu_X^{(\mathrm{M})}(T)}^{(\mathrm{M})} \circ {\mu'}_X^{(\mathrm{M})}. \qquad \square$$

Lemma 4.6 (4.2) *commutes for all* $X \in \mathbf{Set}$.

Proof Let $X \in \mathbf{Set}$ be arbitrary. Writing (4.2) in detail we get



As we have, by definition, that $\mu_x^{(\mathrm{M})} \circ w_{X,T}^{(\mathrm{M})} = v_{X,\mu^{(\mathrm{M})}(T)}^{(\mathrm{M})}$ holds this can be written as



which commutes by the naturality of $\mu^{(\mathrm{M})}$. This shows the claimed statement.$\square$

These lemmata are not just fun facts but will be used in the proof of our main theorem in Chapter 5.

Categorical Characterization of Untyped Lambda Calculus

In this chapter we prove that untyped lambda calculus as defined in Chapter 3 is an initial object in the category of exponential monads. The outline of our proof follows the one in [Zsi06, Chapter 2]. The main difference is that we do all proofs explicitly "by hand", i.e. without any help from computers.

This chapter is divided into two sections. The first one contains the main statement of this thesis and an outline of the proof whereas the second one contains all (technical) proofs of the statements used in the first section.

5.1 Untyped Lambda Calculus as an Initial Object

After all the preparation we are ready to characterize untyped lambda calculus as follows:

Theorem 5.1 *The exponential monad* $(\mathrm{LC}, \mathrm{app1})$ *is an initial object in the category of exponential monads.*

Proof Let $((\mathrm{M}, \eta^{(\mathrm{M})}, \mu^{(\mathrm{M})}), \varepsilon^{(\mathrm{M})})$ be any exponential monad. We need to prove that there is a unique morphism of exponential monads $\varphi : \mathrm{LC} \to \mathrm{M}$. We first show the existence of such a morphism and then its uniqueness.

Existence. We first recursively define a function $\psi_X : \mathrm{SLC}\, X \to \mathrm{M}\, X$ for all $X \in \mathbf{Set}$ and all $L \in \mathrm{SLC}\, X$ by precomposition with C_X of the map given for all $K \in \mathrm{SLC}\, X$ by

$$K \mapsto \begin{cases} \eta_X^{(\mathrm{M})}(x), & \text{if } K = \mathrm{var}_X(x) \text{ with } x \in X, \\ \varepsilon_X^{(\mathrm{M})}(\psi_X(K_1))[* \leftarrow \psi_X(K_2)], & \text{if } K = \mathrm{app}_X(K_1, K_2) \\ & \quad \text{with } L_1, L_2 \in \mathrm{SLC}\, X, \\ {\varepsilon_X^{(\mathrm{M})}}^{-1}(\psi_{X^*}(K')), & \text{if } K = \mathrm{abs}_X(K') \text{ with } K' \in \mathrm{SLC}'\, X, \\ \psi_X(\lambda c.c), & \text{otherwise,} \end{cases}$$

where $c \coloneqq \min\{d \mid d \in D\}$. By precomposition with C_X we always get into one of the first three cases.

By Lemma 5.3 we get that ψ is a natural transformation such that the diagrams corresponding to (2.3) and (2.4) commute. Further we know by Lemma 5.5 that ψ is constant on equivalence classes with respect to α-, β-, and η-equivalence. Therefore we get that the map $\varphi_X : \mathrm{LC}\, X \to \mathrm{M}\, X$ given for all $X \in \mathbf{Set}$ and all

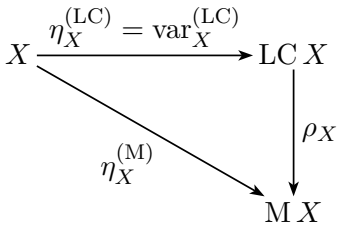
$T \in \mathrm{LC}\, X$ by

$$T \mapsto \psi_X(L),$$

where $L \in \mathrm{SLC}\, X$ is any representative in T, is well-defined. Further it follows that it is even a morphism of monads. Finally by Lemma 5.4 we see that ψ is compatible with $\varepsilon^{(\mathrm{M})}$ and app1. As ψ is compatible with ${\varepsilon^{(\mathrm{M})}}^{-1}$ and abs by definition, ψ being constant on equivalence classes implies that φ is actually a morphism of exponential monads as claimed.

Uniqueness. Let $\rho\colon \mathrm{LC} \dot{\to} \mathrm{M}$ be any morphism of exponential monads. For all $X \in \mathbf{Set}$ and all $T \in \mathrm{LC}\, X$ there exists a representative $L \in \mathrm{SLC}\, X$ of T. We prove by induction that $\rho_X(T) = \psi_X(L)$ holds and therefore $\rho = \varphi$.

If $T = \mathrm{var}_X(x) \in \mathrm{LC}\, X$ for some $x \in X$ we get, by the definition of a morphism of monads, that



commutes. Hence

$$\rho_X(\mathrm{var}_X^{(\mathrm{LC})}(x)) = \eta_X^{(\mathrm{M})}(x) = \psi_X(\mathrm{var}_X^{(\mathrm{SLC})}(x))$$

as needed.

If $T = \mathrm{app}_X(T_1, T_2) \in \mathrm{LC}\, X$ with $T_1, T_2 \in \mathrm{LC}\, X$ we have the following two induction hypotheses:

$$\rho_X(T_i) = \psi_X(L_i)$$

for some (and hence any) $L_i \in \mathrm{SLC}\, X$ belonging to T_i for $i \in \{1, 2\}$. We then need to show that

$$\rho_X(\overline{\mathrm{app}_X(L_1, L_2)}) = \psi_X(\mathrm{app}_X(L_1, L_2))$$

holds, which is equivalent to

$$\rho_X(\overline{\mathrm{app}\, 1_X(L_1)[* \leftarrow L_2]}) = \psi_X(\mathrm{app}\, 1_X(L_1)[* \leftarrow L_2]).$$

By the definition of the equivalence relation as a congruence relation and as morphisms of (exponential) monads commute with substitution (Lemma 4.1) we get

$$\begin{aligned}
\rho_X(\overline{\mathrm{app}\, 1_X(L_1)[* \leftarrow L_2]}) &= \rho_X(\overline{\mathrm{app}\, 1_X(L_1)}[* \leftarrow \overline{L_2}]) \\
&= \rho_{X^*}(\overline{\mathrm{app}\, 1_X(L_1)})[* \leftarrow \rho_X(\overline{L_2})] \\
&= \rho_{X^*}(\mathrm{app}\, 1_X(\overline{L_1}))[* \leftarrow \rho_X(\overline{L_2})].
\end{aligned}$$

By the definition of a morphism of exponential monads applied to ρ and the induction hypotheses we have

$$\begin{aligned}\rho_{X^*}(\mathrm{app}\,1_X(\overline{L_1}))[* \leftarrow \rho_X(\overline{L_2})] &= \varepsilon_X^{(\mathrm{M})}(\rho_X(\overline{L_1}))[* \leftarrow \rho_X(\overline{L_2})] \\ &= \varepsilon_X^{(\mathrm{M})}(\psi_X(L_1))[* \leftarrow \psi_X(L_2)].\end{aligned}$$

Then, as ψ is compatible with $\varepsilon^{(\mathrm{M})}$ and app1 (Lemma 5.4) and commutes with substitution (Lemma 4.1), we conclude with

$$\begin{aligned}\varepsilon_X^{(\mathrm{M})}(\psi_X(L_1))[* \leftarrow \psi_X(L_2)] &= \psi_{X^*}(\mathrm{app1}_X(L_1))[* \leftarrow \psi_X(L_2)] \\ &= \psi_X(\mathrm{app1}_X(L_1)[* \leftarrow \psi_X(L_2)]).\end{aligned}$$

If $T = \mathrm{abs}_X(T') \in \mathrm{LC}\,X$ with $T' \in \mathrm{LC}'\,X$ we have the following induction hypothesis:

$$\rho_{X^*}(T') = \psi_{X^*}(L')$$

for some $L' \in \mathrm{SLC}\,X$ belonging to T'. Our goal is to prove

$$\rho_X(\overline{\mathrm{abs}_X(L')}) = \psi_X(\mathrm{abs}_X(L')).$$

Using the definition of the equivalence relation and the fact that ρ is a morphism of exponential monads it follows that

$$\begin{aligned}\rho_X(\overline{\mathrm{abs}_X(L')}) &= \rho_X(\mathrm{abs}_X(\overline{L'})) \\ &= {\varepsilon_X^{(\mathrm{M})}}^{-1}(\rho_{X^*}(\overline{L'}))\end{aligned}$$

holds. Finally using the induction hypothesis and the definition of ψ we get

$$\begin{aligned}{\varepsilon_X^{(\mathrm{M})}}^{-1}(\rho_{X^*}(\overline{L'})) &= {\varepsilon_X^{(\mathrm{M})}}^{-1}(\psi_{X^*}(L')) \\ &= \psi_X(\mathrm{abs}_X(L')).\end{aligned}$$

□

Corollary 5.2 SLC_δ *for* $\delta \in \{\alpha, \alpha\beta, \alpha\eta\}$ *is characterized as follows:*

(i) SLC_α *is initial among monads with morphisms of modules* $\tau\colon \mathrm{M} \dot{\to} \mathrm{M}'$ *and* $\omega\colon \mathrm{M}' \dot{\to} \mathrm{M}$.

(ii) $\mathrm{SLC}_{\alpha\beta}$ *is initial among monads with morphisms of modules* $\tau : \mathrm{M} \dot{\to} \mathrm{M}'$ *and* $\omega\colon \mathrm{M}' \dot{\to} \mathrm{M}$ *such that* $\tau \circ \omega = \mathrm{Id}_{\mathrm{M}'}$ *holds.*

(iii) $\mathrm{SLC}_{\alpha\eta}$ *is initial among monads with morphisms of modules* $\tau : \mathrm{M} \dot{\to} \mathrm{M}'$ *and* $\omega\colon \mathrm{M}' \dot{\to} \mathrm{M}$ *such that* $\omega \circ \tau = \mathrm{Id}_{\mathrm{M}}$ *holds.*

It is enough to notice that the proof of Theorem (5.1) is modular in the following sense: Whenever we verify a property with respect to β-equivalence we don't assume any properties with respect to η-equivalence and conversely. Further we only use the property that $\varepsilon_X \circ \varepsilon_X^{-1} = \mathrm{Id}_{\mathrm{M}\,X^*}$ holds for all $X \in \mathbf{Set}$ in proofs involving β-equivalence and we only use the property that $\varepsilon_X^{-1} \circ \varepsilon_X = \mathrm{Id}_{\mathrm{M}\,X}$ holds for all $X \in \mathbf{Set}$ when dealing with η-equivalence. By "cutting out" unnecessary parts of the proof above we get the claimed statements.

5.2 Technical Facts

This section only contains (technical) proofs which were omitted in the previous section to yield a nicer presentation of our actual goal. This section is therefore just a list of statements and proofs which is not to be understood as an independent section, i.e. there will be no motivation and/or connection between statements. To keep the statements short we further assume the notation at the point of the previous section at which the statements are claimed to hold.

Lemma 5.3 $\psi : \mathrm{SLC} \to \mathrm{M}$ *is a natural transformation such that the diagrams corresponding to* (2.3) *and* (2.4) *commute..*

Proof Let $f : X \to Y$ be any morphism in **Set**. Proving ψ natural means that we need to show that

$$\begin{array}{ccc} \mathrm{SLC}\, X & \xrightarrow{\psi_X} & \mathrm{M}\, X \\ {\scriptstyle \mathrm{SLC}\, f}\downarrow & & \downarrow{\scriptstyle \mathrm{M}\, f} \\ \mathrm{SLC}\, Y & \xrightarrow[\psi_Y]{} & \mathrm{M}\, Y \end{array}$$

commutes. We prove this by induction on the degree of the λ-terms $L \in \mathrm{SLC}\, X$. For variables we see by the definition of ψ that we need to show

$$\begin{array}{ccc} X & \xrightarrow{\eta_X^{(\mathrm{SLC})}} & \mathrm{M}\, X \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \mathrm{M}\, f} \\ Y & \xrightarrow[\eta_Y^{(\mathrm{SLC})}]{} & \mathrm{M}\, Y \end{array}$$

commutative. This holds due to the fact that $\eta^{(\mathrm{SLC})}$ is a natural transformation.

If $L = \mathrm{app}_X(L_1, L_2) \in \mathrm{SLC}\, X$ with $L_1, L_2 \in \mathrm{SLC}\, X$ we have the following two induction hypotheses:

$$\psi_Y(\mathrm{SLC}\, f(L_i)) = \mathrm{M}\, f(\psi_X(L_i))$$

for $i \in \{1, 2\}$. We need to show that

$$\psi_Y(\mathrm{SLC}\, f(L)) = \mathrm{M}\, f(\psi_X(L))$$

holds. By plugging in the definitions and using the induction hypotheses we get

$$\begin{aligned} \psi_Y(\mathrm{SLC}\, f(L)) &= \psi_Y(\mathrm{app}_Y(\mathrm{SLC}\, f(L_1), \mathrm{SLC}\, f(L_2))) \\ &= \varepsilon_Y^{(\mathrm{M})}(\psi_X(\mathrm{SLC}\, f(L_1)))[* \leftarrow \psi_Y(\mathrm{SLC}\, f(L_2))] \\ &= \varepsilon_Y^{(\mathrm{M})}(\mathrm{M}\, f(\psi_X(L_1)))[* \leftarrow \mathrm{M}\, f(\psi_X(L_2))] \end{aligned}$$

and

$$\mathrm{M}\, f(\psi_X(L)) = \mathrm{M}\, f(\varepsilon_X^{(\mathrm{M})}(\psi_X(L_1))[* \leftarrow \psi_X(L_2)]).$$

Hence we have to show

$$\varepsilon_Y^{(\mathrm{M})}(\mathrm{M}\, f(\psi_X(L_1)))[* \leftarrow \mathrm{M}\, f(\psi_X(L_2))] = \mathrm{M}\, f(\varepsilon_X^{(\mathrm{M})}(\psi_X(L_1))[* \leftarrow \psi_X(L_2)]).$$

This follows from the commutativity of

$$\begin{array}{ccccccc}
\mathrm{M}\, X & \xrightarrow{\varepsilon_X^{(\mathrm{M})}} & \mathrm{M}\, X^* & \xrightarrow{\mathrm{M}\,\gamma_1} & \mathrm{M}\,\mathrm{M}\, X & \xrightarrow{\mu_X^{(\mathrm{M})}} & \mathrm{M}\, X \\
\downarrow{\scriptstyle \mathrm{M}\, f} & & \downarrow{\scriptstyle \mathrm{M}\, f^*} & & \downarrow{\scriptstyle \mathrm{M}\,\mathrm{M}\, f} & & \downarrow{\scriptstyle \mathrm{M}\, f} \\
\mathrm{M}\, Y & \xrightarrow[\varepsilon_Y^{(\mathrm{M})}]{} & \mathrm{M}\, Y^* & \xrightarrow[\mathrm{M}\,\gamma_2]{} & \mathrm{M}\,\mathrm{M}\, Y & \xrightarrow[\mu_Y^{(\mathrm{M})}]{} & \mathrm{M}\, Y,
\end{array}$$

where $\gamma_1 \colon X^* \to \mathrm{M}\, X$ is given for all $z \in X^*$ by

$$z \mapsto \begin{cases} \eta_X^{(\mathrm{M})}(z), & \text{if } z \in X, \\ \psi_X(L_2), & \text{if } z = *\end{cases}$$

and $\gamma_2 \colon Y^* \to \mathrm{M}\, Y$ is given for all $z \in Y$ by

$$z \mapsto \begin{cases} \eta_Y^{(\mathrm{M})}(z), & \text{if } z \in Y, \\ \mathrm{M}\, f(\psi_X(L_2)), & \text{if } z = *. \end{cases}$$

The left square in this diagram commutes as $\varepsilon^{(\mathrm{M})}$ is a natural transformation by definition, the right square commutes as $\mu^{(\mathrm{M})}$ is natural, and the middle square commutes as M is a functor and the square

$$\begin{array}{ccc}
X^* & \xrightarrow{\gamma_1} & \mathrm{M}\, X \\
\downarrow{\scriptstyle f^*} & & \downarrow{\scriptstyle \mathrm{M}\, f} \\
Y^* & \xrightarrow[\gamma_2]{} & \mathrm{M}\, Y
\end{array}$$

commutes: For $x \in X$ this is the same square as the one for the base case and for $*$ we have

$$\mathrm{M}\, f(\gamma_1(*)) = \mathrm{M}\, f(\psi_X(L_2)) = \gamma_2(*) = \gamma_2(f^*(*)),$$

where the "$*$" in the last expression is the one from X^* and the one in the second to last expression is the one from Y^*.

If $L = \mathrm{abs}_X(L') \in \mathrm{SLC}\, X$ with $L' \in \mathrm{SLC}'\, X$ we have the following induction hypothesis:

$$\psi_{Y^*}(\mathrm{SLC}\, f^*(L')) = \mathrm{M}\, f^*(\psi_{X^*}(L')).$$

We have to prove that

$$\psi_Y(\mathrm{SLC}\, f(L)) = \mathrm{M}\, f(\psi_X(L))$$

holds. By plugging in the definitions of $\mathrm{SLC}\, f$ and ψ and using the induction hypothesis we get

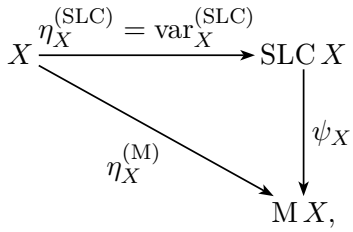
$$\begin{aligned}
\psi_Y(\mathrm{SLC}\, f(L)) &= \psi_Y(\mathrm{SLC}\, f(\mathrm{abs}_X(L'))) \\
&= \psi_Y(\mathrm{abs}_Y(\mathrm{SLC}\, f^*(L'))) \\
&= {\varepsilon_Y^{(\mathrm{M})}}^{-1}(\psi_{Y^*}(\mathrm{SLC}\, f^*(L'))) \\
&= {\varepsilon_Y^{(\mathrm{M})}}^{-1}(\mathrm{M}\, f^*(\psi_{X^*}(L'))).
\end{aligned}$$

By the fact that $\varepsilon^{(\mathrm{M})}$ is natural and the definition of ψ we then get

$$\begin{aligned}
{\varepsilon_Y^{(\mathrm{M})}}^{-1}(\mathrm{M}\, f^*(\psi_{X^*}(L'))) &= \mathrm{M}\, f({\varepsilon_X^{(\mathrm{M})}}^{-1}(\psi_{X^*}(L'))) \\
&= \mathrm{M}\, f(\psi_X(\mathrm{abs}_X(L'))) \\
&= \mathrm{M}\, f(\psi_X(L)).
\end{aligned}$$

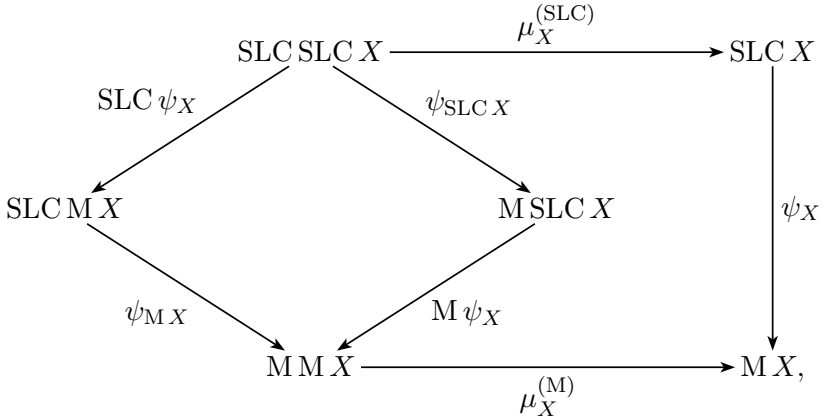
Hence we have shown that ψ is a natural transformation.

The diagram corresponding to (2.3) is



which commutes for all $X \in \mathbf{Set}$ by the definition of ψ on variables.

It is therefore left to show that the diagram corresponding to (2.4), which is



commutes for all $X \in \mathbf{Set}$. First we note that the diamond on the left commutes as we just proved that ψ is a natural transformation. For the rest we are going

to use induction on the outer degree again, as in Lemma 3.2. Let $X \in \mathbf{Set}$ be arbitrary. For the base case we have $L \in \mathrm{SLC}^{\langle 0 \rangle}\,\mathrm{SLC}\,X$, i.e. $L = \mathrm{var}_{\mathrm{SLC}\,X}(K)$ for some $K \in \mathrm{SLC}\,X$. We then get by using the left side of the diamond and the various definitions:

$$\begin{aligned}\mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(L))) &= \mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(\mathrm{var}_{\mathrm{SLC}\,X}(K))))\\ &= \mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{var}_{\mathrm{M}\,X}(\psi_X(K))))\\ &= \mu_X^{(\mathrm{M})}(\eta_{\mathrm{M}\,X}^{(\mathrm{M})}(\psi_X(K))).\end{aligned}$$

By using condition (2.1) of a monad (i.e. that $\eta^{(\mathrm{M})}$ is the unit of the monad) and the definitions again we get that

$$\begin{aligned}\mu_X^{(\mathrm{M})}(\eta_{\mathrm{M}\,X}^{(\mathrm{M})}(\psi_X(K))) &= (\psi_X(K))\\ &= \psi_X(\mu_X^{(\mathrm{SLC})}(\mathrm{var}_{\mathrm{SLC}\,X}(K)))\\ &= \psi_X(\mu_X^{(\mathrm{SLC})}(L))\end{aligned}$$

holds, which finishes the base case.

If $L = \mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2) \in \mathrm{SLC}\,\mathrm{SLC}\,X$ with $L_1, L_2 \in \mathrm{SLC}\,\mathrm{SLC}\,X$ we have the following two induction hypotheses (using the left side of the diamond):

$$\mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(L_i))) = \psi_X(\mu_X^{(\mathrm{SLC})}(L_i))$$

for $i \in \{1, 2\}$. The statement we need to show is

$$\mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(\mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2)))) = \psi_X(\mu_X^{(\mathrm{SLC})}(\mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2))).$$

With the notation from Section 4.2 we can rewrite the second line in the definition of ψ, which was $\varepsilon_X^{(\mathrm{M})}(\psi_X(L_1))[* \leftarrow \psi_X(L_2)]$ (if $L = \mathrm{app}_X(L_1, L_2)$ with $L_1, L_2 \in \mathrm{SLC}\,X$), as $\mathrm{app}_X^{(\mathrm{M})}(\psi_X(L_1), \psi_X(L_2))$. This holds as the maps inducing these maps agree.

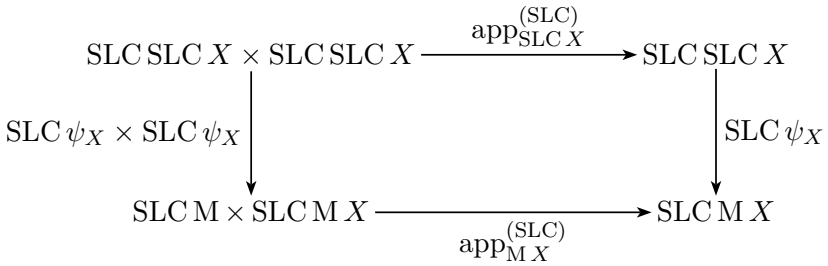
As the two definitions of application for SLC agree, as multiplication commutes with application (Lemma 4.5), and the definition of ψ we have

$$\begin{aligned}\psi_X(\mu_X^{(\mathrm{SLC})}(\mathrm{app}_{\mathrm{SLC}\,X}(L_1, L_2))) &= \psi_X(\mu_X^{(\mathrm{SLC})}(\mathrm{app}_{\mathrm{SLC}\,X}^{(\mathrm{SLC})}(L_1, L_2)))\\ &= \psi_X(\mathrm{app}_X^{(\mathrm{SLC})}(\mu_X^{(\mathrm{SLC})}(L_1), \mu_X^{(\mathrm{SLC})}(L_2)))\\ &= \mathrm{app}_X^{(\mathrm{M})}(\psi_X(\mu_X^{(\mathrm{SLC})}(L_1)), \psi_X(\mu_X^{(\mathrm{SLC})}(L_2))).\end{aligned}$$

Using the induction hypotheses, the commutativity of multiplication with application, and the definition of ψ we derive

$$\begin{aligned}&\mathrm{app}_X^{(\mathrm{M})}(\psi_X(\mu_X^{(\mathrm{SLC})}(L_1)), \psi_X(\mu_X^{(\mathrm{SLC})}(L_2)))\\ &\quad= \mathrm{app}_X^{(\mathrm{M})}(\mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(L_1))), \mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(L_2))))\\ &\quad= \mu_X^{(\mathrm{M})}(\mathrm{app}_{\mathrm{M}\,X}^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(L_1)), \psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(L_2))))\\ &\quad= \mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{app}_{\mathrm{M}\,X}^{(\mathrm{SLC})}(\mathrm{SLC}\,\psi_X(L_1), \mathrm{SLC}\,\psi_X(L_2)))).\end{aligned}$$

As application is a natural transformation



commutes and we can therefore conclude with

$$\begin{aligned}\mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{app}_{\mathrm{M}\,X}^{(\mathrm{SLC})}(\mathrm{SLC}\,\psi_X(L_1),\mathrm{SLC}\,\psi_X(L_2))))\\ = \mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(\mathrm{app}_{\mathrm{SLC}\,X}^{(\mathrm{SLC})}(L_1,L_2))))\\ = \mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(\mathrm{app}_{\mathrm{SLC}\,X}(L_1,L_2)))).\end{aligned}$$

If $L = \mathrm{abs}_{\mathrm{SLC}\,X}(L') \in \mathrm{SLC}\,\mathrm{SLC}\,X$ with $L' \in \mathrm{SLC}'\,\mathrm{SLC}\,X$ we have the following induction hypothesis (using again the left side of the diamond):

$$\mu_{X^*}^{(\mathrm{M})}(\psi_{\mathrm{M}\,X^*}(\mathrm{SLC}\,\psi_{X^*}(\mathrm{SLC}\,g_X^{(\mathrm{SLC})}(L')))) = \psi_{X^*}(\mu_{X^*}^{(\mathrm{SLC})}(\mathrm{SLC}\,g_X^{(\mathrm{SLC})}(L'))),$$

where $g_X^{(\mathrm{SLC})} : (\mathrm{SLC}\,X)^* \to \mathrm{SLC}\,X^*$ is defined by $g_X^{(\mathrm{SLC})}\big|_{\mathrm{SLC}\,X} = \mathrm{SLC}\,\iota_X$ and $g_X^{(\mathrm{SLC})}(*) = \eta_{X^*}^{(\mathrm{SLC})}(*)$. Recall that $g_X^{(\mathrm{SLC})}$ is an injection that does not change the degree of the λ-term, i.e. we have $\deg(L') = \deg(\mathrm{SLC}\,g_X^{(\mathrm{SLC})}(L'))$. We have to prove that

$$\mu_X^{(\mathrm{M})}(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(\mathrm{abs}_{\mathrm{SLC}\,X}(L')))) = \psi_X(\mu_X^{(\mathrm{SLC})}(\mathrm{abs}_{\mathrm{SLC}\,X}(L')))$$

holds. As multiplication and abstraction commute by the definition of a morphism of modules and by the definition of ψ we get

$$\begin{aligned}\psi_X(\mu_X^{(\mathrm{SLC})}(\mathrm{abs}_{\mathrm{SLC}\,X}(L'))) &= \psi_X(\mathrm{abs}_X({\mu'}_X^{(\mathrm{SLC})}(L')))\\ &= {\varepsilon_X^{(\mathrm{M})}}^{-1}(\psi_{X^*}({\mu'}_X^{(\mathrm{SLC})}(L'))).\end{aligned}$$

It then follows by the definition of ${\mu'}_X^{(\mathrm{SLC})}$ and the induction hypothesis, that

$$\begin{aligned}{\varepsilon_X^{(\mathrm{M})}}^{-1}(\psi_{X^*}({\mu'}_X^{(\mathrm{SLC})}(L'))) &= {\varepsilon_X^{(\mathrm{M})}}^{-1}(\psi_{X^*}(\mu_{X^*}^{(\mathrm{SLC})}(\mathrm{SLC}\,g_X^{(\mathrm{SLC})}(L'))))\\ &= {\varepsilon_X^{(\mathrm{M})}}^{-1}(\mu_{X^*}^{(\mathrm{M})}(\psi_{\mathrm{M}\,X^*}(\mathrm{SLC}\,\psi_{X^*}(\mathrm{SLC}\,g_X^{(\mathrm{SLC})}(L'))))).\end{aligned}$$

holds. By the naturality of ψ we have

$$\begin{aligned}\psi_{\mathrm{M}\,X^*} \circ \mathrm{SLC}\,\psi_{X^*} \circ \mathrm{SLC}\,g_X^{(\mathrm{SLC})} &= \psi_{\mathrm{M}\,X^*} \circ \mathrm{SLC}\,g_X^{(\mathrm{M})} \circ \mathrm{SLC}'\,\psi_X\\ &= \mathrm{M}\,g_X^{(\mathrm{M})} \circ \psi_{(\mathrm{M}\,X)^*} \circ \mathrm{SLC}'\,\psi_X,\end{aligned}$$

where $g^{(\mathrm{M})}_X \colon (\mathrm{M}\,X)^* \to \mathrm{M}\,X^*$ is defined analogously to $g^{(\mathrm{SLC})}_X$, which then implies

$$\begin{aligned}\varepsilon^{(\mathrm{M})\,-1}_X(\mu^{(\mathrm{M})}_{X^*}(\psi_{\mathrm{M}\,X^*}(\mathrm{SLC}\,\psi_{X^*}(\mathrm{SLC}\,g^{(\mathrm{SLC})}_X(L')))))\\ = \varepsilon^{(\mathrm{M})\,-1}_X(\mu^{(\mathrm{M})}_X(\mathrm{M}\,g^{(\mathrm{M})}_X(\psi_{(\mathrm{M}\,X)^*}(\mathrm{SLC}'\,\psi_X(L'))))).\end{aligned}$$

By the definition of $\mu'^{(\mathrm{M})}_X$ and the commutativity of multiplication and $\varepsilon^{(\mathrm{M})\,-1}$ we find

$$\begin{aligned}\varepsilon^{(\mathrm{M})\,-1}_X(\mu^{(\mathrm{M})}_X(\mathrm{M}\,g^{(\mathrm{M})}_X(\psi_{(\mathrm{M}\,X)^*}(\mathrm{SLC}'\,\psi_X(L')))))\\ = \varepsilon^{(\mathrm{M})\,-1}_X(\mu'^{(\mathrm{M})}_X(\psi_{(\mathrm{M}\,X)^*}(\mathrm{SLC}'\,\psi_X(L'))))\\ = \mu^{(\mathrm{M})}_X(\varepsilon^{(\mathrm{M})\,-1}_{\mathrm{M}\,X}(\psi_{(\mathrm{M}\,X)^*}(\mathrm{SLC}'\,\psi_X(L'))))).\end{aligned}$$

Finally by the definition of ψ and the naturality of abstraction we get

$$\begin{aligned}\mu^{(\mathrm{M})}_X(\varepsilon^{(\mathrm{M})\,-1}_{\mathrm{M}\,X}(\psi_{(\mathrm{M}\,X)^*}(\mathrm{SLC}'\,\psi_X(L')))) &= \mu^{(\mathrm{M})}_X(\psi_{\mathrm{M}\,X}(\mathrm{abs}_X(\mathrm{SLC}'\,\psi_X(L'))))\\ &= \mu^{(\mathrm{M})}_X(\psi_{\mathrm{M}\,X}(\mathrm{SLC}\,\psi_X(\mathrm{abs}_{\mathrm{SLC}\,X}(L')))),\end{aligned}$$

which finishes the inductive step and the proof. □

Lemma 5.4 *$\psi \colon \mathrm{SLC} \to \mathrm{M}$ is compatible with $\varepsilon^{(\mathrm{M})}$ and* app1*, i.e. for all $X \in$ **Set** and $L \in \mathrm{SLC}\,X$ it holds that*

$$\psi_{X^*}(\mathrm{app1}_X(L)) = \varepsilon^{(\mathrm{M})}_X(\psi_X(L)).$$

Proof Let $X \in$ **Set** be arbitrary. We need to prove

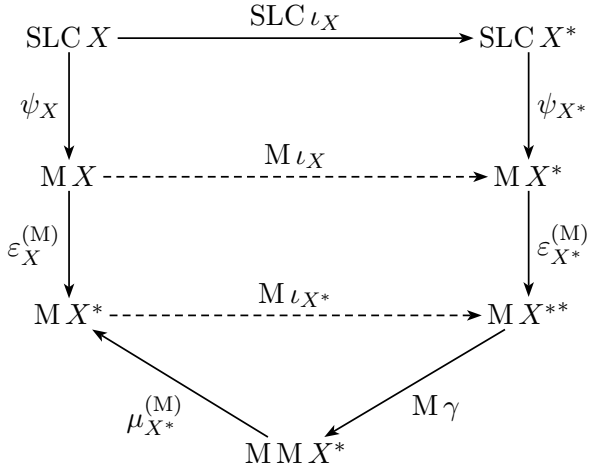
$$\begin{array}{ccc}\mathrm{SLC}\,X & \xrightarrow{\psi_X} & \mathrm{M}\,X\\ \Big\downarrow{\scriptstyle \mathrm{app1}_X} & & \Big\downarrow{\scriptstyle \varepsilon^{(\mathrm{M})}_X}\\ \mathrm{SLC}'\,X & \xrightarrow[\psi_{X^*}]{} & \mathrm{M}'\,X\end{array}$$

commutative. By plugging in the definitions of $\mathrm{app1}_X$ and ψ_{X^*} we need to show that

$$\varepsilon^{(\mathrm{M})}_{X^*}(\psi_{X^*}(L))[\star \leftarrow \psi_{X^*}(*)] = \psi_X(\varepsilon^{(\mathrm{M})}_X(L))$$

holds for all $L \in \mathrm{SLC}\,X$, where we use $\star$ to denote the element for the disjoint union with X^* to distinguish it from $*$, i.e. $X^{**} = (X \amalg *) \amalg \star$. In other words

we need to prove the outside of



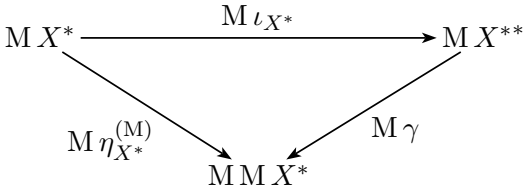
commutative, where $\gamma\colon X^{**} \to \mathrm{M}\, X^*$ is given for all $y \in X^{**}$ by

$$y \mapsto \begin{cases} \eta^{(\mathrm{M})}_{X^*}(y), & \text{if } y \in X^*, \\ \eta^{(\mathrm{M})}_{X^*}(*) = \psi_{X^*}(*), & \text{if } y = \star. \end{cases}$$

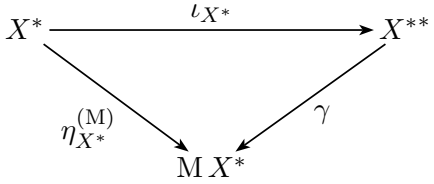
The upper rectangle in this diagram commutes as ψ is a natural transformation (proof of Lemma 5.3) and the lower rectangle commutes as $\varepsilon^{(\mathrm{M})}$ is a natural transformation by the definition of an exponential monad (the reader may want to redraw the rectangle using that $\mathrm{M}\, X^* = \mathrm{M}'\, X$ holds). Hence the statement follows by proving that

$$\mu^{(\mathrm{M})}_{X^*} \circ \mathrm{M}\, \gamma \circ \mathrm{M}\, \iota_{X^*} = \mathrm{Id}_{\mathrm{M}\, X^*}$$

holds. By condition (2.1) of a monad we know that $\mathrm{M}\, \eta^{(\mathrm{M})}_{X^*} \circ \mu^{(\mathrm{M})}_{X^*} = \mathrm{Id}_{\mathrm{M}\, X^*}$ holds, therefore we are done by proving



commutative. However this is obvious as M is a functor and as by the definition of γ the triangle



commutes. □

Lemma 5.5 *Let $X \in \mathbf{Set}$ be any set and let $K, L \in \mathrm{SLC}\, X$ be any λ-terms such that $\overline{K} = \overline{L}$ holds in $\mathrm{LC}\, X$. Then it holds that $\psi_X(K) = \psi_X(L)$.*

Proof As in Lemma 3.2 it is sufficient to prove that the statement holds for any two λ-terms which differ by one step of any of the three equivalences.

Given the assumptions of the lemma, α-equivalence is again no problem due to precomposition with C_X.

For β-equivalence assume that $L \in \mathrm{SLC}\, X$ contains a β-redex, i.e. is of the form $\mathrm{app}_X(\mathrm{abs}_X(L_1), L_2)$ with $L_1 \in \mathrm{SLC}'\, X$ and $L_2 \in \mathrm{SLC}\, X$. We then get the following chain of equalities:

$$\begin{aligned}
\psi_X(\mathrm{app}_X(\mathrm{abs}_X(L_1), L_2)) &= \varepsilon^{(\mathrm{M})}_X(\psi_X(\mathrm{abs}_X(L_1)))[* \leftarrow \psi_X(L_2)] \\
&= \varepsilon^{(\mathrm{M})}_Y(\varepsilon^{(\mathrm{M})^{-1}}_X(\psi_{X^*}(L_1)))[* \leftarrow \psi_X(L_2)] \\
&= \psi_{X^*}(L_1)[* \leftarrow \psi_X(L_2)] \\
&= \psi_X(L_1[* \leftarrow \psi_X(L_2)]).
\end{aligned}$$

The first two equations hold by the definition of ψ, the third one due to the fact that $\varepsilon^{(\mathrm{M})}_Y \circ \varepsilon^{(\mathrm{M})^{-1}}_X = \mathrm{Id}_{\mathrm{M}\, X^*}$ holds, and the last one holds by Lemma 4.1.

For η-equivalence we assume that $L \in \mathrm{SLC}\, X$ contains an η-redex, i.e. is of the form $\mathrm{abs}_X(\mathrm{app}_{X^*}(\mathrm{SLC}\, \iota_X(K), \mathrm{var}_{X^*}(*)))$ with $K \in \mathrm{SLC}\, X$. We then get the following chain of equalities:

$$\begin{aligned}
\psi_X(\mathrm{abs}_X(\mathrm{app}_{X^*}(\mathrm{SLC}\, \iota_X(K), \mathrm{var}_{X^*}(*)))) &= \psi_X(\mathrm{abs}_X(\mathrm{app1}_X(K))) \\
&= \varepsilon^{(\mathrm{M})^{-1}}_X(\psi_{X^*}(\mathrm{app1}_X(K))) \\
&= \varepsilon^{(\mathrm{M})^{-1}}_X(\varepsilon^{(\mathrm{M})}_X(\psi_X(K))) \\
&= \psi_X(K).
\end{aligned}$$

In this case we used the definition of $\mathrm{app1}_X$ for the first equation, the definition of ψ for the second one, Lemma 5.4 for the third one, and $\varepsilon^{(\mathrm{M})^{-1}}_X \circ \varepsilon^{(\mathrm{M})}_X = \mathrm{Id}_{\mathrm{M}\, X}$ for the last one. □

Chapter 6

Outlook and Summary

In this chapter we give some ideas of how one can continue with the results of the previous chapters at hand and also mention alternative approaches. In Section 6.1 we describe how one can deduce similar results for simply typed lambda calculus. We don't give full details but mention some important points. Afterwards we give some ideas we did not have time to work out in the limited time allotted for a master's thesis in Section 6.2. Some ideas may not be complete and some guesses may even turn out to be wrong. This is of course not a complete list of possible topics but rather a starting point for further research.

6.1 Simply Typed Lambda Calculus

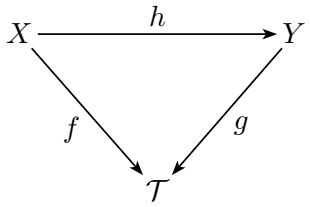
In this section we describe how one can transform the previous chapters to yield a similar result for simply typed lambda calculus. Giving all details would be cumbersome and not very enlightening, we therefore only give some key ideas. The notation of this chapter is based on [HM10] and the outline loosely follows [Zsi06].

For the rest of this section let $\mathcal{B}$ be any non-empty set (whose elements are called *base types*). The set $\mathcal{T}$ of all *types* with respect to $\mathcal{B}$ is defined by:

- Whenever $t \in \mathcal{B}$, then $t \in \mathcal{T}$.
- Whenever $s, t \in \mathcal{T}$, then $s \to t \in \mathcal{T}$.
- Only the elements defined above belong to $\mathcal{T}$.

Any family of sets indexed by $\mathcal{T}$ is referred to as a *$\mathcal{T}$-set*. For any $\mathcal{T}$-set X and any $x \in X$ we denote by $x : t$ that x belongs to the component of type t in X. Equivalently we can regard $\mathcal{T}$-sets as objects in the slice category $(\mathbf{Set} \downarrow \mathcal{T})$. In the following we work with both points of view and pick the one that feels more intuitive to us in the given situation.

The *morphisms* of $\mathcal{T}$-sets are morphism in $(\mathbf{Set} \downarrow \mathcal{T})$, i.e. type preserving maps: Given $f\colon X \to \mathcal{T}, g\colon Y \to \mathcal{T} \in (\mathbf{Set} \downarrow \mathcal{T})$ a morphism between f and g is a map $h\colon X \to Y$ such that



commutes.

For all $t \in \mathcal{T}$ we denote by $\tau_t \colon (\mathbf{Set} \downarrow \mathcal{T}) \to \mathbf{Set}$ the functor that assigns to any $\mathcal{T}$-set X the component of type t, denoted by X_t. For any $\mathcal{T}$-set X we have $X = \amalg_{t \in \mathcal{T}} X_t$. In case that X already has a lower index we add the type to this index separated by a semicolon. By $\jmath_{X_t}$ we denote the map that includes X_t into X such that all elements of X_t are of type t.

Before we continue we need to generalize the notion of a (right) module. Let $\mathscr{C}$ be any category and let (T, η, μ) be any monad over $\mathscr{C}$. A *(right)* T*-module* is a pair (M, σ), where $\mathrm{M} \colon \mathscr{C} \to \mathscr{D}$ is a functor and $\sigma \colon \mathrm{M}\,\mathrm{T} \dot{\to} \mathrm{M}$ is a natural transformation such that the diagrams corresponding to (2.5) commute.

A monad over $(\mathbf{Set} \downarrow \mathcal{T})$ is called a $\mathcal{T}$*-monad*. Let T be any $\mathcal{T}$-monad and let $(\mathrm{M}, \sigma^{(\mathrm{M})})$ with $\mathrm{M} \colon (\mathbf{Set} \downarrow \mathcal{T}) \to (\mathbf{Set} \downarrow \mathcal{T})$ be any T-module. We then have for all $t \in \mathcal{T}$ a module $(\mathrm{M}_t, \sigma^{(\mathrm{M}_t)})$, where $\mathrm{M}_t \colon (\mathbf{Set} \downarrow \mathcal{T}) \to \mathbf{Set}$ is given by

$$\mathrm{M}_t \coloneqq \tau_t \,\mathrm{M}$$

and $\sigma^{(\mathrm{M}_t)} \colon \mathrm{M}_t\,\mathrm{T} = \tau_t\,\mathrm{M}\,\mathrm{T} \dot{\to} \mathrm{M}_t$ is given for all $\mathcal{T}$-sets X by

$$\sigma^{(\mathrm{M}_t)}_X \coloneqq \tau_t(\sigma^{(\mathrm{M})}_X(\jmath_{(\mathrm{M}\,\mathrm{T}\,X)_t}(\mathrm{M}\,\mathrm{T}\,X)_t)).$$

More intuitively this means applying $\sigma^{(\mathrm{M})}$ as if we regard elements of $\tau_t\,\mathrm{M}\,\mathrm{T}\,X$ as elements of $\mathrm{M}\,\mathrm{T}\,X$. This holds as $\sigma^{(\mathrm{M})}$ is a natural transformation, hence its components preserve types.

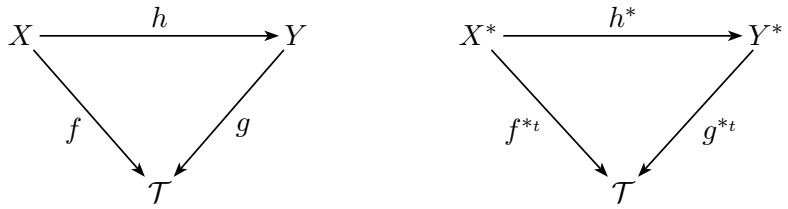
In contrast to modules over a monad over **Set** we don't have just one derived module but a derived module for every type $t \in \mathcal{T}$. More precisely let T be any $\mathcal{T}$-monad and let M be any T-module. For all $t \in \mathcal{T}$ we denote by $\partial_t M$ the *derived module of* M *with respect to* t, which is given for all $f \colon X \to \mathcal{T} \in (\mathbf{Set} \downarrow \mathcal{T})$ by

$$\partial_t\,\mathrm{M}\,f \coloneqq \mathrm{M}\,f^{*t},$$

where $f^{*t} \colon X^* \to \mathcal{T}$ is given by $f^{*t}|_X = f$ and $f^{*t}(*) = t$, and for all morphisms $h \in (\mathbf{Set} \downarrow \mathcal{T})$ by

$$\partial_t\,\mathrm{M}\,h \coloneqq \mathrm{M}\,h^*.$$

This definition can be visualized as first transforming the left triangle in



into the right one and then applying M. In the following a notation of the form X^{*t} for some $\mathcal{T}$-set X and some $t \in \mathcal{T}$ means that we add the singleton $*$ of type t to X.

All proofs we referenced in Chapter 2, e.g. that the derived module is again a module over the same monad, still hold in the same form. It is enough to note that all maps involved are actually morphisms of the slice category. Therefore the untyped case implies the commutativity of the needed diagrams and the morphisms imply compatibility with the types. For example let (T, η, μ) be any $\mathcal{T}$-monad. For all $\mathcal{T}$-sets X and all $t \in \mathcal{T}$ the map $g_{X;t} \colon (\mathrm{T}\, X)^{*_t} \to \mathrm{T}\, X^{*_t}$ given by $g_{X;t}|_{\mathrm{T}\, X} = \mathrm{T}\, \iota_X$ and $g_{X;t}(*) = \eta_{X^{*_t}}(*)$ is a morphism of $\mathcal{T}$-sets as ι_X is a morphism, T is a functor, and $*$ is of type t in both cases.

An *exponential $\mathcal{T}$-monad* is a $\mathcal{T}$-monad M with, for all $s, t \in \mathcal{T}$, an isomorphism $\varepsilon^{(\mathrm{M})}_{s,t} \colon \mathrm{M}_{s\to t} \to \partial_s \mathrm{M}_t$. These monads together with the obvious morphism form a category as in Lemma 2.2.

We now define a functor $\mathrm{SST} \colon (\mathbf{Set} \downarrow \mathcal{T}) \to (\mathbf{Set} \downarrow \mathcal{T})$ similar to SLC. Let $D_{\mathcal{T}}$ be a fixed $\mathcal{T}$-set where every component has the same properties as the set D in the untyped case. We then define SST on objects for all $\mathcal{T}$-sets X by

$$X \mapsto \{L \mid L \text{ is a } \mathcal{T}\text{-typed } \lambda\text{-term with variables in } X \amalg D_{\mathcal{T}} \text{ with } \mathrm{FV}(L) \subseteq X\},$$

where the disjoint union of X and $D_{\mathcal{T}}$ is given componentwise:

$$X \amalg D_{\mathcal{T}} = \amalg_{t\in\mathcal{T}}(X_t \amalg D_{\mathcal{T};t}).$$

The typing environment Γ_X for any $\mathcal{T}$-sets X is given by

$$\begin{aligned}
(\text{bound})&\frac{d : t \in D_{\mathcal{T}}}{\Gamma_X \vdash \mathrm{var}(d) : t}\\
(\text{var})&\frac{x : t \in X}{\Gamma_X \vdash \mathrm{var}(x) : t}\\
(\text{app})&\frac{\Gamma_X \vdash L_1 : s \to t \quad \Gamma_X \vdash L_2 : s}{\Gamma_X \vdash L_1 L_2 : t}\\
(\text{abs})&\frac{\Gamma_X \vdash L : t \quad \Gamma_X \vdash x : s}{\Gamma_X \vdash \lambda x.L : s \to t}.
\end{aligned}$$

We can then define a canonical form similar to the untyped case (the renaming of the variables has to respect their type). For application we need to note that this is no longer a morphism of SST-modules $\mathrm{SST} \times \mathrm{SST} \to \mathrm{SST}$ as we need to regard the types. However we still have morphisms of SST-modules $\mathrm{app}_{s,t} \colon \mathrm{SST}_{s\to t} \times \mathrm{SST}_s \to \mathrm{SST}_t$ for all $s, t \in \mathcal{T}$. It is important to note that we regard SST_t as an object in **Set** for all $t \in \mathcal{T}$ - if we were regarding it as an object in $(\mathbf{Set} \downarrow \mathcal{T})$ with all elements of the same type the product $\mathrm{SST}_{s\to t} \times \mathrm{SST}_s$ would be empty (recall that the product in $(\mathbf{Set} \downarrow \mathcal{T})$ is determined by the pullback in **Set**). Similarly we have for all $s, t \in \mathcal{T}$ that abstraction is a morphism of SST-modules $\mathrm{abs}_{s,t} \colon \partial_s \mathrm{SST}_t \to \mathrm{SST}_{s\to t}$.

With these remarks we can then define the degree of a typed λ-term and the action of SST on morphisms analogously to the untyped case by first mapping

the terms to the module corresponding to their types, then applying the maps described above, and afterwards including them back into the monad.

Further SST_δ and ST are defined analogously to SLC_δ and LC respectively for all $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$. These are again monads when the unit and the multiplication are defined as in the untyped case. We also have that ST is an exponential $\mathcal{T}$-monad with $\mathrm{abs}_{s,t} = \varepsilon_{s,t}^{(\mathrm{ST})^{-1}} : \partial_s \mathrm{ST}_t \to \mathrm{ST}_{s\to t}$ whose inverse $\mathrm{app1}_{s,t} \colon \mathrm{ST}_{s\to t} \to \partial_s \mathrm{ST}_t$ is induced by

$$\mathrm{app1}_{X;s,t}(L) \coloneqq \mathrm{app}_{X^{*s};s,t}(\mathrm{SST}\,\iota_X(L), \mathrm{var}_{X^{*s}}(*))$$

for all $\mathcal{T}$-sets X, all $s, t \in \mathcal{T}$, and all $L \in \mathrm{SST}_{s\to t}\, X$.

Almost all of our proofs in the previous chapters were elementary in the sense that we always reduced the problem at hand to its simplest case, e.g. when proving a map constant with respect to some equivalence relation we showed that that map is constant for one step of the generating relation. Therefore the proofs can be generalized by replacing statements of the form "for all $X \in \mathbf{Set}$ and all $L \in \mathrm{SLC}\, X$" by "for all $\mathcal{T}$-sets X, all $t \in \mathcal{T}$, and all $L \in \mathrm{SLC}_t\, X$" and accordingly changing the indices of the maps involved. This also holds for our general principle of proofs by induction on the degree of λ-terms.

We also see that our general notion of substitution (Equation (3.9)) is still well-defined as T_2 and x need to be of the same type by the definition of substitution in simply typed lambda calculus.

In the generalization of application we need to pay more attention. We therefore give all details of its definition. For any $\mathcal{T}$-monad M with morphisms of modules $\varepsilon_{s,t}^{(\mathrm{M})} \colon \mathrm{M}_{s\to t} \to \partial_s \mathrm{M}_t$ we define for all $\mathcal{T}$-sets X and Y and all $s, t \in \mathcal{T}$ a map $\mathrm{app}_{X,Y;s,t}^{(\mathrm{M})} \colon \mathrm{M}_{s\to t}\, X \times \mathrm{M}_s\, Y \to \mathrm{M}_t(X \amalg Y)$ by the following chain of maps:

$$\begin{aligned}\mathrm{M}_{s\to t}\, X \times \mathrm{M}_s\, Y &\xrightarrow{\varepsilon_{X;s,t}^{(\mathrm{M})} \times \mathrm{Id}_{\mathrm{M}_s\, Y}} \mathrm{M}_t\, X^{*s} \times \mathrm{M}_s\, Y \xrightarrow{k_{X,Y;s,t}^{(\mathrm{M})}} \mathrm{M}_t(X \amalg \mathrm{M}\, Y) \xrightarrow{l_{X,Y;s,t}^{(\mathrm{M})}} \cdots \\ &\cdots \xrightarrow{l_{X,Y;s,t}^{(\mathrm{M})}} \mathrm{M}_t\, \mathrm{M}(X \amalg Y) \xrightarrow{\mu_{X\amalg Y}^{(\mathrm{M}_t)}} \mathrm{M}_t(X \amalg Y).\end{aligned}$$

The maps used are the following:

- $k_{X,Y;s,t}^{(\mathrm{M})} \colon \mathrm{M}_t\, X^{*s} \times \mathrm{M}_s\, Y \to \mathrm{M}_t(X \amalg \mathrm{M}\, Y)$ is determined by

$$r_{X,Y;s,t}^{(\mathrm{M})} \colon \mathrm{M}_s\, Y \to \mathrm{Hom}_{(\mathbf{Set}\downarrow\mathcal{T})}(X^{*s}, X \amalg \mathrm{M}\, Y)$$
$$T \mapsto \begin{cases} x \mapsto x, & \text{if } x \in X, \\ * \mapsto T, \end{cases}$$

  which then induces a map from $\mathrm{M}_s\, Y$ to $\mathrm{Hom}_{(\mathbf{Set}\downarrow\mathcal{T})}(\mathrm{M}\, X^{*s}, \mathrm{M}(X \amalg \mathrm{M}\, Y))$. However $\mathrm{Hom}_{(\mathbf{Set}\downarrow\mathcal{T})}(A, B)$ for any $\mathcal{T}$-sets A and B is itself a $\mathcal{T}$-set by

$$(\mathrm{Hom}_{(\mathbf{Set}\downarrow\mathcal{T})}(A, B))_t \coloneqq \{f \mid f \colon A_t \to B_t\} = \mathrm{Hom}_{\mathbf{Set}}(A_t, B_t)$$

for all $t \in \mathcal{T}$. Hence we get a map

$$\mathrm{M}_s\, Y \to \mathrm{Hom}_{\mathbf{Set}}(\mathrm{M}_t\, X^{*_s}, \mathrm{M}_t(X \amalg \mathrm{M}\, Y)),$$

which determines $k^{(\mathrm{M})}_{X,Y;s,t}$.

- $l^{(\mathrm{M})}_{X,Y;s,t} \colon \mathrm{M}_t(X \amalg \mathrm{M}\, Y) \to \mathrm{M}_t\, \mathrm{M}(X \amalg Y)$ is induced by

$$s^{(\mathrm{M})}_{X,Y;s,t} \colon X \amalg \mathrm{M}\, Y \to \mathrm{M}(X \amalg Y),$$
$$z \mapsto \begin{cases} \eta^{(\mathrm{M})}_{X \amalg Y}(z), & \text{if } z \in X, \\ \mathrm{M}\, j_Y(z), & \text{if } z \in \mathrm{M}\, Y. \end{cases}$$

  $s^{(\mathrm{M})}_{X,Y;s,t}$ is obviously a morphism of $\mathcal{T}$-sets as $\eta^{(\mathrm{M})}$ is natural and j_Y is a morphism of $\mathcal{T}$-sets by definition.

- $\mu^{(\mathrm{M}_t)}$ is derived from $\mu^{(\mathrm{M})}$ in the same way as $\sigma^{(\mathrm{M}_t)}$ was derived from $\sigma^{(\mathrm{M})}$.

The maps $\mathrm{app}^{(\mathrm{M})}_{X;s,t} \colon \mathrm{M}_{s \to t}\, X \times \mathrm{M}_s\, X \to \mathrm{M}_t\, X$ for all $\mathcal{T}$-sets X and all $s, t \in \mathcal{T}$ are then defined as in the untyped case.

This leads to the analog statement of Theorem 5.1, which is:

Theorem 6.1 *The exponential $\mathcal{T}$-monad* $(\mathrm{ST}, \{\mathrm{app1}_{s,t}\}_{s,t \in \mathcal{T}})$ *is an initial object in the category of exponential $\mathcal{T}$-monads.*

Of course one also gets the statements corresponding to the ones in Corollary 5.2.

6.2 Miscellaneous

In this section we give some ideas of what else could have been done or what could have been done in another way. We don't give full details in this section as its intention is to show where one could go in a next step.

6.2.1 Cartesian Closed Categories

Here we follow the ideas of [Sco80]. Let $\mathscr{C}$ be any cartesian closed category with small hom-sets. For all objects $D \in \mathscr{C}$ we get an endofunctor on **Set** defined for all $X \in \mathbf{Set}$ by

$$\mathrm{M}_D\, X \coloneqq \mathrm{Hom}_{\mathscr{C}}(D^X, D)$$

and for all morphisms $f : X \to Y$ in **Set** by precomposition with the morphism induced by the universal property of the product and the morphisms $\prod_{y \in Y} D_y \xrightarrow{\pi_y} D_y \cong D_x$ whenever $f(x) = y$, where π_y denotes the projection morphism of the product.

This endofunctor can be turned into a monad in the following way: The unit is given for all $X \in \mathbf{Set}$ by

$$\begin{aligned} \eta_X \colon X &\to \mathrm{Hom}_{\mathscr{C}}(D^X, D) \\ x &\mapsto \prod_{x \in X} D_x \xrightarrow{\pi_x} D_x \cong D \end{aligned}$$

and the multiplication $\mu_X \colon \mathrm{Hom}_{\mathscr{C}}(D^{\mathrm{Hom}_{\mathscr{C}}(D^X, D)}, D) \to \mathrm{Hom}_{\mathscr{C}}(D^X, D)$ is defined for all $X \in \mathbf{Set}$ by precomposition with the morphism from D^X to $D^{\mathrm{Hom}_{\mathscr{C}}(D^X, D)}$ which is induced by the universal property of the product and the morphisms $D^X \xrightarrow{\varphi} D \cong D_\varphi$ for all $\varphi \in \mathrm{Hom}_{\mathscr{C}}(D^X, D)$.

Assume that the object $D \in \mathscr{C}$ is such that there exists an isomorphism $D \cong D^D$. In this case the monad M_D is exponential, where the isomorphism is induced by

$$\begin{aligned} \mathrm{M}_D\, X^* &= \mathrm{Hom}_{\mathscr{C}}(D^{X \amalg *}, D) \\ &\cong \mathrm{Hom}_{\mathscr{C}}(D^X \times D, D) \\ &\cong \mathrm{Hom}_{\mathscr{C}}(D^X, D^D) \\ &\cong \mathrm{Hom}_{\mathscr{C}}(D^X, D). \end{aligned}$$

Theorem 5.1 implies that there exists a unique morphism of exponential monads $\mathrm{LC} \to \mathrm{M}_D$. By weakening the assumption on the object D in the obvious ways one gets into settings where it is possible to apply Corollary 5.2.

Guess 6.2 Let $\mathscr{C}$ be a cartesian closed category with small hom-sets generated by one object D and morphisms $\varepsilon : D \to D^D$ and $\varepsilon^{-1} : D^D \to D$ such that $\varepsilon \circ \varepsilon^{-1} = \mathrm{Id}_{D^D}$ and $\varepsilon^{-1} \circ \varepsilon = \mathrm{Id}_D$ hold. Then $\mathrm{M}_D \cong \mathrm{LC}$.

6.2.2 Operadic Approach

In this subsection we consider lambda calculus from an operadic point of view. All ideas in this part as well as the next subsection are due to my advisor.

In the following we give a short definition of an operad in **Set**. For a more "down to earth" introduction we refer the reader to the literature (consider for example [Fre12], which contains a nice and general description).

An S-module is a sequence of sets $\mathcal{X} = (\mathcal{X}(n))_{n \in \mathbb{N}}$ together with a right group action of S_n on $\mathcal{X}(n)$ for all $n \in \mathbb{N}$, where S_n denotes the symmetric group on n elements. A morphism between S-modules $\mathcal{X}$ and $\mathcal{Y}$ is a sequence of equivariant maps $f = (f_n \colon \mathcal{X}(n) \to \mathcal{Y}(n))_{n \in \mathbb{N}}$. S-modules together with the corresponding morphisms form a category which we denote by S- mod.

Let G be any group and let X and Y be any sets with a right action of G. By composing with the inverse operation of G we get a left action of G on Y. We then define

$$X \underset{G}{\times} Y \coloneqq X \times Y / \{((xg, y), (x, gy)) \mid x \in X, y \in Y, g \in G\}.$$

This allows us to define a monoidal structure $\circ$ on S- mod given for all objects $\mathcal{X}, \mathcal{Y} \in S$- mod and all $n \in \mathbb{N}$ by

$$(\mathcal{X} \circ \mathcal{Y})(n) \coloneqq \coprod_{\substack{k, m_1 + \cdots + m_k = n \\ k, m_1, \ldots, m_k \in \mathbb{N}}} (\mathcal{Y}(k) \underset{S_k}{\times} (\mathcal{X}(m_1) \times \cdots \times \mathcal{X}(m_k))) \underset{S_{m_1} \times \cdots \times S_{m_k}}{\times} S_n.$$

The unit $\mathbb{1} = (\mathbb{1}(n))_{n \in \mathbb{N}}$ is given by

$$\mathbb{1}(n) = \begin{cases} *, & \text{if } n = 1, \\ \emptyset, & \text{otherwise.} \end{cases}$$

An *operad* in **Set** is a monoid in the monoidal category $(S\text{- mod}, \circ)$.

Let $\mathcal{O}$ be an operad. A *right $\mathcal{O}$-module* is an object $\mathcal{P}$ in S- mod that is a right module over the monoid $\mathcal{O}$ with respect to the monoidal structure $\circ$. If $\mathcal{P}$ is a right $\mathcal{O}$-module we define its derivative $\mathcal{P}'$ by

$$\mathcal{P}'(n) \coloneqq \mathcal{P}(n+1)$$

for all $n \in \mathbb{N}$, where the action of S_n comes from the inclusion $S_n \hookrightarrow S_{n+1}$ induced by $i_n \colon \{1, \ldots, n\} \hookrightarrow \{1, \ldots, n+1\}$ defined by $i(l) = l$ for all $l \in \{1, \ldots, n\}$, and the operad structure is given similarly for all $k, m_1, \ldots, m_k \in \mathbb{N}$ by

$$\begin{aligned} \mathcal{P}'(k) \underset{S_k}{\times} \mathcal{O}(m_1) \times \cdots \times \mathcal{O}(m_k) & \\ &= \mathcal{P}(k+1) \underset{S_{k+1}}{\times} \mathcal{O}(m_1) \times \cdots \times \mathcal{O}(m_k) \\ &\to \mathcal{P}(k+1) \underset{S_{k+1}}{\times} \mathcal{O}(m_1) \times \cdots \times \mathcal{O}(m_k) \times \mathcal{O}(1) \\ &\to \mathcal{P}(m_1 + \cdots + m_k + 1) \\ &= \mathcal{P}'(m_1 + \cdots + m_k). \end{aligned}$$

An *exponential operad* is an operad $\mathcal{O}$ together with an isomorphism of right $\mathcal{O}$-modules $\varepsilon \colon \mathcal{O} \xrightarrow{\sim} \mathcal{O}'$.

Lambda calculus (without equivalence relations) can be seen as an operad $\lambda\,\mathrm{Calc}$ by setting

$$\lambda\,\mathrm{Calc}(n) \coloneqq \left\{ L \;\middle|\; \begin{array}{l} L \text{ is a } \lambda\text{-term with variables in } \{x_1, \ldots, x_n\} \amalg D \text{ with} \\ \mathrm{FV}(L) = \{x_1, \ldots, x_n\} \text{ and every free variable occurs} \\ \text{exactly once} \end{array} \right\}$$

for all $n \in \mathbb{N}$, where D is the same set as in the previous chapters. The action of S_n is the permutation of the n variables. The operad structure is induced for all $k, m_1, \ldots, m_k \in \mathbb{N}$ by

$$\begin{gathered} \lambda\,\mathrm{Calc}(k) \times \lambda\,\mathrm{Calc}(m_1) \times \cdots \times \lambda\,\mathrm{Calc}(m_k) \to \lambda\,\mathrm{Calc}(m_1 + \cdots + m_k) \\ (T, (T_1, \ldots, T_k)) \mapsto (T[x_i \leftarrow T_i \mid i \in \{1, \ldots, n\}])_{\mathrm{C}_{\{x_1, \ldots, x_{m_1 + \cdots + m_k}\}}}, \end{gathered}$$

where we implicitly rename the free variables in the λ-terms T_i for $i \in \{1, \dots, n\}$ accordingly. We further define two morphisms app1 : $\lambda\,\mathrm{Calc} \to \lambda\,\mathrm{Calc}'$ and abs: $\lambda\,\mathrm{Calc}' \to \lambda\,\mathrm{Calc}$ for all $n \in \mathbb{N}$ by

$$\begin{aligned} \mathrm{app1}_n\colon \quad & \lambda\,\mathrm{Calc}(n) \to \lambda\,\mathrm{Calc}'(n) \\ & T \mapsto \mathrm{app}(T, x_{n+1}), \end{aligned}$$

$$\begin{aligned} \mathrm{abs}_n\colon \quad & \lambda\,\mathrm{Calc}'(n) \to \lambda\,\mathrm{Calc}(n) \\ & T \mapsto (\lambda c.(T[x_{n+1} \leftarrow c]))_{\mathrm{C}_{\{x_1,\dots,x_n\}}}, \end{aligned}$$

with $c \coloneqq \min\{d \mid d \in D \text{ and } d \notin \mathrm{FV}(T)\}$.

Guess 6.3 The following statements hold:

(i) $\lambda\,\mathrm{Calc}$ is initial among operads $\mathcal{O}$ together with morphisms of $\mathcal{O}$-modules $\varepsilon\colon \mathcal{O} \to \mathcal{O}'$ and $\varepsilon^{-1}\colon \mathcal{O}' \to \mathcal{O}$.

(ii) app1 and abs descend to the quotients $\lambda\,\mathrm{Calc}_\delta$ for $\delta \in \{\alpha, \alpha\beta, \alpha\eta, \alpha\beta\eta\}$, where $\lambda\,\mathrm{Calc}_\delta$ is defined analogously to SLC_δ.

(iii) $\lambda\,\mathrm{Calc}_{\alpha\beta\eta}$ is an exponential operad.

(iv) $\lambda\,\mathrm{Calc}_{\alpha\beta\eta}$ is initial among exponential operads.

Proof (Sketch of (i)) We just give the main idea of the proof. We are less rigorous than in the rest of this thesis. More precisely we don't mention assumptions, conditions, or quantifiers when understood and we also use notation that should intuitively be clear without an explicit definition.

Let $\mathcal{O}$ be any operad with morphisms of right $\mathcal{O}$-modules $\varepsilon : \mathcal{O} \to \mathcal{O}'$ and $\varepsilon^{-1} : \mathcal{O}' \to \mathcal{O}$. Define a morphism of S-modules $\psi\colon \lambda\,\mathrm{Calc} \to \mathcal{O}$ inductively on the degree of the λ-terms by:

- $\psi^{\langle 0\rangle} : \lambda\,\mathrm{Calc}^{\langle 0\rangle} \to \mathcal{O}$ is defined as the unit of $\mathcal{O}$. This comes from the fact that the inclusion $\lambda\,\mathrm{Calc}^{\langle 0\rangle} \hookrightarrow \lambda\,\mathrm{Calc}$ is the unit of $\lambda\,\mathrm{Calc}$.
- Note that we have by the definitions that

  $$\mathrm{app}(L', L'') = \mathrm{app1}(L')[x_{\deg(L')+1} \leftarrow L''] = \mathrm{app1}(L') \circ (\mathrm{Id}, \dots, \mathrm{Id}, L'')$$

  holds whenever any (and hence all) of the expressions are well-defined.

  If $L = \mathrm{app}(L', L'') \in \lambda\,\mathrm{Calc}^{\langle k\rangle}$ for some $k \in \mathbb{N} \setminus \{0\}$ we define

  $$\psi(L) \coloneqq \varepsilon(\psi(L')) \circ (\mathrm{Id}, \dots, \mathrm{Id}, \psi(L'')).$$

  For convenience we denote this by $\mathrm{app}(\psi(L'), \psi(L''))$.
- If $L = \lambda c.(L'[x_{n+1} \leftarrow c]) \in \lambda\,\mathrm{Calc}^{\langle k\rangle}$ for some $k \in \mathbb{N} \setminus \{0\}$ we set

  $$\psi(L) \coloneqq \varepsilon^{-1}(\psi(L')).$$

  This implies that the corresponding morphisms commute with ψ.

One can check that ψ is well-defined and a morphism of S-modules. Further it commutes with the given module morphisms.
Claim 1: ψ is a morphism of operads.

We need to prove that $\psi(L \circ (L_1, \dots, L_n)) = \psi(L) \circ (\psi(L_1), \dots, \psi(L_n))$ holds (whenever either of the two expressions makes sense). We use induction on the the degree of the λ-term L:

- If $L \in \lambda\,\mathrm{Calc}^{\langle 0 \rangle}$ the property is satisfied as ψ is defined as the unit of $\mathcal{O}$ on $\lambda\,\mathrm{Calc}^{\langle 0 \rangle}$.
- If $L = \mathrm{app}(L', L'') \in \lambda\,\mathrm{Calc}$ we get by Claim 2 below (for the second and fifth equations), the definition of ψ (for the third and sixth equations), and the induction hypothesis that

$$\begin{aligned}
&\psi(\mathrm{app}(L', L'') \circ (L_1, \dots, L_n)) \\
&\quad = \psi(\mathrm{app}(L' \circ (L_1, \dots, L_m), L'' \circ (L_{m+1}, \dots, L_n)) \\
&\quad = \mathrm{app}(\psi(L' \circ (L_1, \dots, L_m)), \psi(L'' \circ (L_{m+1}, \dots, L_n))) \\
&\quad = \mathrm{app}(\psi(L') \circ (\psi(L_1), \dots, \psi(L_m)), \psi(L'') \circ (\psi(L_{m+1}), \dots, \psi(L_n))) \\
&\quad = \mathrm{app}(\psi(L'), \psi(L'')) \circ (\psi(L_1), \dots, \psi(L_n)) \\
&\quad = \psi(\mathrm{app}(L', L'')) \circ (\psi(L_1), \dots, \psi(L_n))
\end{aligned}$$

  holds, which finishes this case.
- If $L = \mathrm{abs}(L')$ for some $L' \in \lambda\,\mathrm{Calc}^{\langle k-1 \rangle}(n+1)$ we have

$$\begin{aligned}
\psi(\mathrm{abs}(L') \circ (L_1, \dots, L_n)) &= \psi(\mathrm{abs}(L' \circ (L_1, \dots, L_n, \mathrm{Id}))) \\
&= \varepsilon^{-1}(\psi(L' \circ (L_1, \dots, L_n, \mathrm{Id}))) \\
&= \varepsilon^{-1}(\psi(L') \circ (\psi(L_1), \dots, \psi(L_n), \mathrm{Id})) \\
&= \varepsilon^{-1}(\psi(L')) \circ (\psi(L_1), \dots, \psi(L_n))
\end{aligned}$$

  by using that abs is a morphism of $\lambda\,\mathrm{Calc}$-modules, the definition of ψ, the induction hypothesis, and that ε^{-1} is a morphism of $\mathcal{O}$-modules.

Claim 2: If $\mathcal{O}$ is an operad and $\varepsilon\colon \mathcal{O} \to \mathcal{O}'$ is a morphism of $\mathcal{O}$-modules, then

$$\mathrm{app}(L', L'') \circ (L_1, \dots, L_n) = \mathrm{app}(L' \circ (L_1, \dots, L_m), L'' \circ (L_{m+1}, \dots, L_n))$$

holds (whenever either of the two expressions makes sense).

We get the result by the definitions and the fact that ε is a morphism of $\mathcal{O}$-modules:

$$\begin{aligned}
&\mathrm{app}(L', L'') \circ (L_1, \dots, L_n) \\
&\quad = \varepsilon(L') \circ (\mathrm{Id}, \dots, \mathrm{Id}, L'') \circ (L_1, \dots, L_n) \\
&\quad = \varepsilon(L') \circ (L_1, \dots, L_m, \mathrm{Id}) \circ (\mathrm{Id}, \dots, \mathrm{Id}, L'' \circ (L_{m+1}, \dots, L_n)) \\
&\quad = \varepsilon(L' \circ (L_1, \dots, L_m)) \circ (\mathrm{Id}, \dots, \mathrm{Id}, L'' \circ (L_{m+1}, \dots, L_n)) \\
&\quad = \mathrm{app}(L' \circ (L_1, \dots, L_m), L'' \circ (L_{m+1}, \dots, L_n)).
\end{aligned}$$

□

6.2.3 Connection of the Two Approaches

In this final subsection we give some connections between the monadic and the operadic approach. This subsection is mostly a collection of ideas that we did not have enough time to work out in detail. The structure is therefore not perfect but we welcome the reader to read it nonetheless as we point out where connections between several constructions we used might be found.

Convention We assume (implicitly) that all categories in this subsection have small hom-sets.

From Cartesian Closed Categories to Operads

Let $\mathscr{C}$ be a cartesian closed category and let D be an object therein. The *endomorphism operad* $\mathcal{End}_{\mathscr{C}}(D)$ *of* D is defined for all $n \in \mathbb{N}$ by

$$(\mathcal{End}_{\mathscr{C}}(D))(n) \coloneqq \mathrm{Hom}_{\mathscr{C}}(D^n, D).$$

The right action of S_n on $\mathrm{Hom}_{\mathscr{C}}(D^n, D)$ is given for all $n \in \mathbb{N}$ by the left action of S_n on D^n. The operad structure is induced for all $k, m_1, \ldots, m_k \in \mathbb{N}$ by

$$\begin{array}{c} \mathrm{Hom}_{\mathscr{C}}(D^k, D) \times \mathrm{Hom}_{\mathscr{C}}(D^{m_1}, D) \times \cdots \times \mathrm{Hom}_{\mathscr{C}}(D^{m_k}, D) \\ \downarrow \mathrm{Id}_{\mathrm{Hom}_{\mathscr{C}}(D^k, D)} \times (k\text{-fold product of maps}) \\ \mathrm{Hom}_{\mathscr{C}}(D^k, D) \times \mathrm{Hom}_{\mathscr{C}}(D^{m_1+\cdots+m_k}, D^k) \\ \downarrow \text{composition} \\ \mathrm{Hom}_{\mathscr{C}}(D^{m_1+\cdots+m_k}, D). \end{array}$$

Given an operad $\mathcal{O}$ and a cartesian closed category $\mathscr{C}$ we define an $\mathcal{O}$*-algebra* as an object $D \in \mathscr{C}$ with a morphism of operads $\rho_D \colon \mathcal{O} \to \mathcal{End}_{\mathscr{C}}(D)$. $\mathcal{O}$-algebras with their structure preserving morphisms define a category which we denote by $\mathcal{O}$-alg.

Note that for $\mathcal{O} = \mathcal{End}_{\mathscr{C}}(D)$ the functor M_D defined in Subsection 6.2.1 can be extended to a functor from **Set** to $\mathcal{O}$-alg as we have an obvious morphism of operads

$$(\rho_{\mathrm{Hom}_{\mathscr{C}}(D^Y, D)})_n \colon \mathrm{Hom}_{\mathscr{C}}(D^n, D) \to \mathrm{Hom}_{\mathbf{Set}}(\mathrm{Hom}_{\mathscr{C}}(D^Y, D)^n, \mathrm{Hom}_{\mathscr{C}}(D^Y, D))$$

for all $Y \in \mathbf{Set}$ and all $n \in \mathbb{N}$. One might further try to characterize this functor to establish a connection to the construction in Subsection 6.2.1.

From Operads to Monads

Using the notation of the previous paragraph we have an obvious forgetful functor $\mathrm{G}: \mathcal{O}\text{-alg} \to \mathscr{C}$. If this functor has a left adjoint (which it does for example if $\mathscr{C} = \mathbf{Set}$) we call it the *free $\mathcal{O}$-algebra functor*. This adjunction defines a monad $\mathrm{M}_{\mathcal{O}}: \mathscr{C} \to \mathscr{C}$.

A *reflexive* (respectively a *coreflexive*) operad is an operad $\mathcal{O}$ together with morphisms of $\mathcal{O}$-modules $\varepsilon: \mathcal{O} \to \mathcal{O}'$ and $\varepsilon^{-1}: \mathcal{O}' \to \mathcal{O}$ such that $\varepsilon \circ \varepsilon^{-1} = \mathrm{Id}_{\mathcal{O}'}$ ($\varepsilon^{-1} \circ \varepsilon = \mathrm{Id}_{\mathcal{O}}$ respectively) holds. There are statements corresponding to the last two points of Guess 6.3 for reflexive and coreflexive operads. Further we also assume:

Guess 6.4 If $\mathcal{O}$ is an exponential (respectively reflexive or coreflexive) operad, then $\mathrm{M}_{\mathcal{O}}$ is an exponential (respectively reflexive or coreflexive) monad.

From Monads to Cartesian Closed Categories

By combining the previous two paragraphs we get a functor from cartesian closed categories to monads. Our goal is now to construct a left adjoint to this functor.

Let (M, η, μ) be any monad over **Set**. We construct a category $\mathscr{C}_{\mathrm{M}}$ as follows:

- The objects of $\mathscr{C}_{\mathrm{M}}$ are the objects of **Set**.
- The hom-sets are given for all objects $X, Y \in \mathscr{C}_{\mathrm{M}}$ by

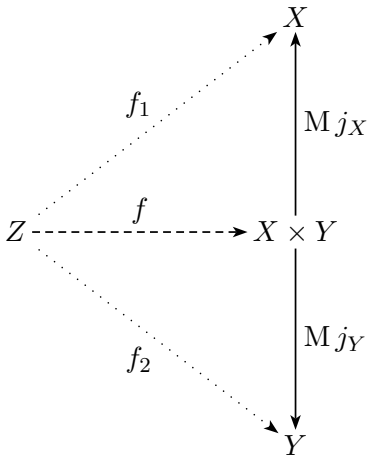
$$\mathrm{Hom}_{\mathscr{C}_{\mathrm{M}}}(X, Y) \coloneqq \mathrm{Hom}_{\mathbf{Set}}(Y, \mathrm{M}\,X).$$

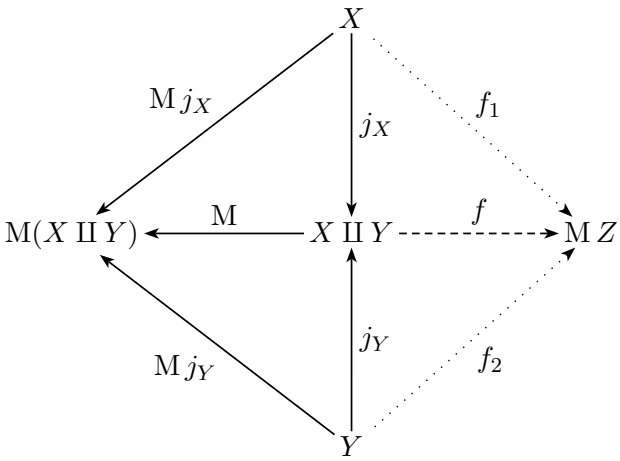
- Composition of morphisms is given by the multiplication of M. More precisely we define the composition of $f : Y \to \mathrm{M}\,X \in \mathrm{Hom}_{\mathscr{C}_{\mathrm{M}}}(X, Y)$ and $g: Z \to \mathrm{M}\,Y \in \mathrm{Hom}_{\mathscr{C}_{\mathrm{M}}}(Y, Z)$ for any $X, Y, Z \in \mathscr{C}_{\mathrm{M}}$ by

$$g \circ f \coloneqq Z \xrightarrow{g} \mathrm{M}\,Y \xrightarrow{\mathrm{M}\,f} \mathrm{M}\,\mathrm{M}\,X \xrightarrow{\mu_X} \mathrm{M}\,X \in \mathrm{Hom}_{\mathscr{C}_{\mathrm{M}}}(X, Z).$$

- The identity morphism is given by the unit of M, i.e. for all $X \in \mathscr{C}_{\mathrm{M}}$ we have $\mathrm{Id}_X = \eta_X: X \to \mathrm{M}\,X \in \mathrm{Hom}_{\mathscr{C}_{\mathrm{M}}}(X, X)$.
- The cartesian structure on $\mathscr{C}_{\mathrm{M}}$ is given by the disjoint union of sets. In detail let $X, Y \in \mathscr{C}_{\mathrm{M}}$ be arbitrary. The construction is visualized in the diagrams below, where we depict the situation in $\mathscr{C}_{\mathrm{M}}$ in the top diagram and the situation in **Set** in the lower one. Starting in the bottom diagram we have the solid arrows given by the disjoint union, M, and composition. The composition maps are then the projections of the product in $\mathscr{C}_{\mathrm{M}}$. For the universal property assume we are given $Z \in \mathscr{C}_{\mathrm{M}}$ with morphisms $f_1: Z \to X$ and $f_2: Z \to Y$ (dotted arrows). These correspond to maps $X \to \mathrm{M}\,Z$ and $Y \to \mathrm{M}\,Z$ in **Set**. Therefore by the universal property of $X \amalg Y$ there exists a unique map $f: X \amalg Y \to \mathrm{M}\,Z$ (dashed arrow) such that the bottom diagram commutes. By the definition of morphisms in $\mathscr{C}_{\mathrm{M}}$ this is the unique

morphism such that the top diagram commutes.





For the one-element set $* \in \mathscr{C}_{\mathrm{M}}$ we have that the endofunctor M_* is isomorphic to M as we have

$$\mathrm{M}_* Y = \mathrm{Hom}_{\mathscr{C}_{\mathrm{M}}}(*^Y, *) \cong \mathrm{Hom}_{\mathscr{C}_{\mathrm{M}}}(Y, *) = \mathrm{Hom}_{\mathbf{Set}}(*, \mathrm{M}\, Y) \cong \mathrm{M}\, Y$$

for all $Y \in \mathbf{Set}$.

The composed functor from monads to cartesian closed categories to monads is isomorphic to the identity. On the other hand if we start with a cartesian closed category $\mathscr{C}$ and an object $D \in \mathscr{C}$ we have a functor from $\mathscr{C}_{\mathrm{M}_D}$ to $\mathscr{C}$ which is given for all objects $X \in \mathscr{C}_{\mathrm{M}_D}$ by $X \mapsto D^X$ and on morphisms by

$$\begin{aligned}\mathrm{Hom}_{\mathscr{C}_{\mathrm{M}_D}}(X, Y) &= \mathrm{Hom}_{\mathbf{Set}}(Y, \mathrm{M}_D\, X) = \mathrm{Hom}_{\mathbf{Set}}(Y, \mathrm{Hom}_{\mathscr{C}}(D^X, D)) \\ &\cong \mathrm{Hom}_{\mathscr{C}}(D^X, D^Y).\end{aligned}$$

We therefore have a natural transformation to the identity from which we can then conclude that we indeed have an adjunction.

From Monads to Operads

Next we construct a left adjoint to the functor that sends an operad to the monad defined be the free algebra adjunction.

Let (M, η, μ) be any monad over **Set**. We construct an operad $\mathcal{O}_\mathrm{M}$ as follows:

- Set $\mathcal{O}_\mathrm{M}(n) \coloneqq \mathrm{M}[n]$ for all $n \in \mathbb{N}$, where $[n] \coloneqq \{1, \dots, n\}$. The action of S_n comes from the functoriality of M.
- The operad structure is induced for all $k, m_1, \dots, m_k \in \mathbb{N}$ by

$$
\begin{array}{c}
\mathrm{M}[k] \times \mathrm{M}[m_1] \times \cdots \times \mathrm{M}[m_k] \\
\downarrow{\scriptstyle \mathrm{Id}_{\mathrm{M}[k]} \times \mathrm{M}([k_i] \hookrightarrow [m_1 + \cdots + m_k])} \\
\mathrm{M}[k] \times (\mathrm{M}[m_1 + \cdots + m_k])^k \\
\downarrow{\scriptstyle \mathrm{Id}_{\mathrm{M}[k]} \times \cong} \\
\mathrm{M}[k] \times \mathrm{Hom}_{\mathbf{Set}}([k], \mathrm{M}[m_1 + \cdots + m_k]) \\
\downarrow{\scriptstyle \mathrm{Id}_{\mathrm{M}[k]} \times \mathrm{M}} \\
\mathrm{M}[k] \times \mathrm{Hom}_{\mathbf{Set}}(\mathrm{M}([k]), \mathrm{M}\,\mathrm{M}[m_1 + \cdots + m_k]) \\
\downarrow{\scriptstyle \text{evaluation}} \\
\mathrm{M}\,\mathrm{M}[m_1 + \cdots + m_k] \\
\downarrow{\scriptstyle \mu_{[m_1 + \cdots + m_k]}} \\
\mathrm{M}[m_1 + \cdots + m_k].
\end{array}
$$

The obvious maps from $\mathcal{O}(n)$ to $\mathrm{M}_\mathcal{O}[n]$ lead to a natural transformation from the identity on the category of operads to the composite of the two functors defined above. On the other hand we have for all monads (M, η, μ) and all $X \in \mathbf{Set}$ that $\mathrm{M}\,X$ is an $\mathcal{O}_\mathrm{M}$-algebra, where

$$\mathcal{O}_\mathrm{M}(n) = \mathrm{M}[n] \to \mathrm{Hom}_{\mathbf{Set}}((\mathrm{M}\,X)^n, \mathrm{M}\,X)$$

is given as above. By adjunction the unit $\eta_X \colon X \to \mathrm{M}\,X$ determines a morphism of $\mathcal{O}_\mathrm{M}$-algebras from $\mathrm{M}_{\mathcal{O}\mathrm{M}}X$ to $\mathrm{M}\,X$. This leads to the second natural transformation of the adjunction from which we can then conclude that we actually constructed a left adjoint.

Adjunctions and Modules

Recall that a monad over **Set** can be described as a monoid in the monoidal category $(\mathbf{Set}^{\mathbf{Set}}, \circ)$, where $\mathbf{Set}^{\mathbf{Set}}$ denotes the category of all covariant functors from **Set** to **Set**. We then have an adjunction

$$S\text{-}\operatorname{mod} \underset{\mathrm{G}}{\overset{\mathrm{F}}{\rightleftarrows}} \mathbf{Set}^{\mathbf{Set}}.$$

F is given by the Schur functor, i.e. we define

$$\begin{aligned} \mathrm{F}\colon S\text{-}\operatorname{mod} &\to \mathbf{Set}^{\mathbf{Set}} \\ \mathcal{X} &\mapsto \left(Y \mapsto \mathrm{S}(\mathcal{X}, Y) \coloneqq \coprod_{n\in\mathbb{N}} \mathcal{X}(n) \underset{S_n}{\times} Y^n \right). \end{aligned}$$

Its right adjoint G is given by

$$\begin{aligned} \mathrm{G}\colon \mathbf{Set}^{\mathbf{Set}} &\to S\text{-}\operatorname{mod} \\ \mathrm{M} &\mapsto \left(\mathrm{G}\,\mathrm{M}(n) \coloneqq \operatorname{Hom}_{\mathbf{Set}^{\mathbf{Set}}}(\mathrm{P}^{(n)}, \mathrm{M}) \right)_{n\in\mathbb{N}}, \end{aligned}$$

where $\mathrm{P}\colon \mathbf{Set} \to \mathbf{Set}$ is defined for all $Y \in \mathbf{Set}$ by $\mathrm{G}\,Y \coloneqq Y^n$.

Let $\mathcal{O}$ be any operad and let $\mathrm{M}_{\mathcal{O}}$ be the associated monad. The $\mathrm{M}_{\mathcal{O}}$-module $\mathrm{M}_{\mathcal{O}'}$ is *not* isomorphic to $\mathrm{M}'_{\mathcal{O}}$. However we still have a natural transformation $\mathrm{M}_{\mathcal{O}'} \to \mathrm{M}'_{\mathcal{O}}$. This leads to the following statement:

Guess 6.5 If $\mathcal{O}$ is initial among exponential operads, then $\mathrm{M}_{\mathcal{O}}$ is initial among exponential monads.

As we don't have the other direction of this statement we can not give an equivalence between the monadic approach and the operadic approach but we get at least some connection between them.

6.3 Summary

Chapters 3 through 5 were devoted to prove the main statement of this thesis which is Theorem 5.1. This proof turned out to be longer than expected and (unfortunately) also very technical without a lot of insight. We hope to have provided at least some kind of motivation for our approach.

Due to that fact the time left for generalizations or other approaches was very limited. But we still managed to give some examples thereof in the previous sections of this chapter. First we showed how one can derive a similar statement for simply typed lambda calculus. One could now proceed by trying to generalize this to other types of lambda calculus. Afterwards we gave the main idea for an approach via operads. Finally we shed some light on the connection between the two approaches that could also be deepened in a further step.

Bibliography

[Bar84] Henk Barendregt. *The Lambda Calculus – It's Syntax and Semantics*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. Elsevier Science Publishers B.V. (North-Holland), 1984.

[BB00] Henk Barendregt and Erik Barendsen. Introduction to Lambda Calculus. Lecture Notes, 2000.

[Fre12] Benoit Fresse. Operads & Grothendieck-Teichmüller Groups. Draft Document, 2012.

[HM04] André Hirschowitz and Marco Maggesi. Modules over monads and typoids, 2004.

[HM05] André Hirschowitz and Marco Maggesi. Modules over Monads, Monadic Syntax and the Category of Untyped Lambda-Calculi, 2005.

[HM10] André Hirschowitz and Marco Maggesi. Modules over monads and initial semantics. *Information and Computation*, 208(5):545 – 564, 2010. Special Issue: 14th Workshop on Logic, Language, Information and Computation (WoLLIC 2007).

[ML98] Saunders Mac Lane. *Categories for the Working Mathematician*. Graduate Texts in Mathematics. Springer, 1998.

[Sco80] D. S. Scott. Relating theories of the λ-calculus. In *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 403–450. Academic Press, 1980.

[Sel07] Peter Selinger. Lecture Notes on the Lambda Calculus. Lecture Notes, 2007.

[Zsi06] Julianna Zsidó. Le lambda calcul vu comme monade initiale. Master's thesis, Université de Nice – Laboratoire J. A. Dieudonné, 2005/06. Mémoire de Recherche – master 2.