Treball final de grau

GRAU DE MATEMÀTIQUES

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

A categorical view of algebraic theories

Autor: Aina Ferrà Marcús

Director: Dr. Carles Casacuberta
Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, June 27, 2018

Abstract

Classically, algebraic structures such as groups, rings, and many others were jointly studied with the language of universal algebra. It was later found that certain tools from category theory, called *monads*, are especially suitable to encode the whole amount of information contained in algebraic theories.

In this work we discuss monads, and, in particular, some monads that are relevant in functional programming in Computer Science. We give a proof of the equivalence between the category of algebraic theories (formalized as *Lawvere theories*) and the category of finitary monads on the category of sets. We also prove that there is an equivalence between the category of algebras over a monad and the category of models of the associated Lawvere theory. Finally, we apply this equivalence of categories to give a new proof of the fact that all localizations on the category of abelian groups can be uniquely lifted to R-modules for every ring R.

Resum

Les estructures algebraiques com grups, anells i moltes altres es poden tractar en un context comú que clàssicament s'anomenava àlgebra universal. En teoria de categories, les *mònades* són endofunctors que resulten ser adients per sintetitzar tota la informació continguda en les teories algebraiques.

En aquest treball s'estudien les mònades i, en particular, alguns exemples de mònades que són rellevants en la programació funcional en Informàtica. Després es dona una demostració de l'equivalència entre la categoria de les teories algebraiques (formalitzada com *teories de Lawvere*) i la categoria de les mònades finitàries a la categoria dels conjunts. També es demostra que hi ha una equivalència entre la categoria d'àlgebres sobre una mònada donada i la categoria de models de la teoria de Lawvere associada a aquesta mònada. Finalment s'aplica aquesta equivalència de categories per donar una nova demostració del fet que totes les localitzacions a la categoria dels grups abelians s'eleven de manera única als R-mòduls per a qualsevol anell R.

2010 Mathematics Subject Classification. 18C15, 18C10, 03C05

Acknowledgements

First, I would like to thank my professor Dr. Carles Casacuberta, for being more than a guidance to me. Thank you for showing me the beauty that lies in topology; thank you for being patient, providing me this opportunity and not letting me jump off the ship.

I would like to thank my two best friends Jose and Ignasi. You have been with me for our whole undergraduate time and you have made the long lasting afternoons in the university much better.

Thanks to the first mentor I had, Marco, for showing me that to know something is not the same as to prove something.

I would like to thank my partner in crime Nico. Thank you for telling me that my work looked cool even though neither you nor me understood it.

And of course, thanks to my family. You may not understand the intricate ways of Mathematics, but you are always there for me.

Contents

1 Introduction

Universal algebra is the study of *algebraic structures*, such as groups and rings, by focusing on their properties instead of their elements. In universal algebra, one takes an algebraic structure and studies it in terms of operations and relations. In 1898, A. N. Whitehead published *A Treatise on Universal Algebra*, where he credited W. R. Hamilton and A. De Morgan as the originators of the subject matter, as well as J. J. Sylvester with coining the term itself. At the time, structures such as Lie algebras or hyperbolic quaternions drew attention to the need of expanding algebraic structures beyond the typical associatively multiplicative ones. Besides that, G. Boole's algebra of logic made a strong counterpoint to the ordinary number algebra, so the term *universal* encoded a larger field of Mathematics.

The work on this subject was minimal until the early 1930s, when G. Birkhoff and O. Ore began publishing about it. In the following years, more papers on the matter were published, dealing with free algebras, congruence and subalgebra lattices, and homomorphism theorems. Some development of mathematical logic was also made in the 1940s by A. Maltsev, although it went unnoticed due to the war. In 1963, with his thesis, W. F. Lawvere built a bridge between category theory and universal algebra.

In universal algebra, an *algebra* or *algebraic structure* is a set A together with a collection of operations on A. An n-ary operation is a function that takes n elements of A and returns a single element of A. Thus, a 0-ary operation (also called *nullary* operation) is represented as an element of A (a *constant*). A 2-ary operation (also called *binary* operation) is often denoted by a symbol between its arguments. Operations of higher arity are usually denoted by function symbols. After the operations have been specified, the nature of the algebra is further defined by axioms, which in universal algebra take the form of equational laws. A collection of algebraic structures defined by equational laws is called a *variety* or an *equational class*.

Let us take groups as an example. A group G consists of a set of elements together with a binary operation $x \cdot y$ with the following properties:

- associativity: for all x, y, z in G we have $x \cdot (y \cdot z) = (x \cdot y) \cdot z$;
- identity element: there exists an element e in G such that for all x in G we have $e \cdot x = x \cdot e = x$;
- existence of inverses: for every x in G there exists an element x^{-1} such that $x \cdot x^{-1} = x^{-1} \cdot x = e$.

In universal algebra, this concept can be translated with some small differences. A group is a set G together with operations $\cdot\colon G^2 \to G$, called *multiplication*; $e\colon G^0 \to G$, called *identity*; and $(\ )^{-1} : G \to G$, called *inverse*, such that the following equational laws hold:

- for all x, y, z in G, we have $x \cdot (y \cdot z) = (x \cdot y) \cdot z$;

- for all x in G, we have $e \cdot x = x \cdot e = x$;
- for all x in G we have $x \cdot x^{-1} = x^{-1} \cdot x = e$.

The main difference between those concepts is that, in the common definition, the identity and inverse elements are defined element-wise, i.e., they are *quantified* laws. In universal algebra there are no quantified laws and all of the properties are inherent to the nature of the operations. Although this may seem like a technical difference, it has immediate practical consequences in category theory: when defining a group object in category theory (where the object need not be a set), one must use equational laws rather than quantified laws, as objects in general categories need not have elements. Besides, universal algebra insists on the idea that inverse and identity are maps —a strong and useful idea in category theory.

Lawvere was responsible for translating this concept into the categorical world. A *Lawvere theory* or *algebraic theory* is a finite-product-preserving contravariant functor I from the category of finite cardinals to a small category $\mathcal{L}$ whose objects are the finite cardinals, and I is assumed to be the identity on objects. This accounts for the varieties in universal algebra. To define specific examples of algebraic structures, the concept of a *model* arises. A model of a Lawvere theory is a finite-product-preserving functor M from the category $\mathcal{L}$ of the given theory to any other category $\mathcal{C}$ (usually the category of sets).

Retaking our group example, one can define the *theory of groups* as a Lawvere theory where the small category $\mathcal{L}$ is the category whose morphisms are defined as

$$\mathcal{L}(n, m) = \mathsf{Grp}(F_m, F_n),$$

where Grp is the category of groups and F_n is the free group with n generators. To define a group in this setting, pick any model $M\colon \mathcal{L} \to \mathsf{Set}$ and consider the set $G = M(1)$ with the structure imposed by M. For example, let us find the multiplication in G. Consider the morphism

$$m\colon 2 \to 1$$

in $\mathcal{L}$ corresponding to the homomorphism $t \mapsto xy$ from the free group $F_1 = F(t)$ on one generator to the free group $F_2 = F(x, y)$ on two generators. Since M preserves finite products, we obtain a map

$$G^2 = G \times G = M(1) \times M(1) \cong M(2) \to M(1) = G,$$

where the last arrow is chosen to be $M(m)$. This defines indeed a multiplication map in G. The same method can be used to obtain the identity map and the inversion map. This illustrates the similarity with universal algebra: inverses and identity are not defined element-wise, but by means of maps.

Lawvere theories are not the only categorical way to look at universal algebra. Equational classes can also be represented by means of *monads*. A monad on a category $\mathcal{C}$ is a functor $T\colon \mathcal{C} \to \mathcal{C}$ together with two natural transformations $\eta\colon \mathrm{Id}_{\mathcal{C}} \to T$ (called *unit*) and $\mu\colon TT \to T$ called (*multiplication*) such that certain properties

are met (see Section 3). The connection with universal algebra becomes clear when looking at monads from another point of view, namely adjunctions. Every adjunction yields a monad and every monad can be retrieved from an adjunction, in fact in more than one way. One way to convert a monad T into an adjunction is through the *Eilenberg-Moore category* of T, also called the category of T*-algebras*. A T-algebra in $\mathcal{C}$ is a pair (A, a) consisting of an object A of $\mathcal{C}$ and a morphism $a\colon TA \to A$ such that some extra properties are met, as made precise in Section 3.

The intuitive idea is that an algebraic structure can be thought of a set with additional structure that depends on the operations defined. Let us consider, for example, the adjunction

$$F\colon \mathsf{Set} \rightleftarrows \mathsf{Grp} : U,$$

where the free functor F returns, for a set X, the free group generated by X, and the forgetful functor U returns the underlying set of a given group. We can consider the corresponding monad $T = UF$, which, given a set X, returns the set of formal words on X. In this case, a T-algebra $(X,\, a\colon TX \to X)$ defines an instance of a group, since a group structure on a set X corresponds precisely to a retraction of the free group FX onto the set X, that is, a function $FX \to X$ whose composite with the canonical map $X \to FX$ is the identity on X. In this example, it is important to note that the free group FX is a directed union of finitely generated free groups $F(x_1, \ldots, x_n)$, namely those generated by finite subsets of X. As we shall see, it turns out that in all adjunctions associated with Lawvere theories the left adjoint F has this property, namely that its value on a set X is determined by the values on finite subsets of X. The resulting monad $T = UF$ is then called *finitary*, which means that it commutes with directed unions.

Hence, both Lawvere theories and monads translate the concept of universal algebra to the categorical world, so it is natural to expect that there is a strong relation between them. This work focuses on the equivalence between the category Law of Lawvere theories and the category $\mathsf{FinMonSet}$ of finitary monads on sets. For a Lawvere theory L, we denote by $\mathsf{Mod}\, L$ the category of models of L in Set. Then we can associate to L a monad T_L that comes from the adjunction

$$F_L\colon \mathsf{Set} \rightleftarrows \mathsf{Mod}L : U$$

where U returns $M(1)$ for every model $M\colon \mathcal{L} \to \mathsf{Set}$, and F_L is its left adjoint. Conversely, given a monad T, we associate to it a Lawvere theory L_T by defining its morphisms as

$$\mathcal{L}_T(n, m) = \mathrm{Kl}(T)(m, n)$$

where $\mathrm{Kl}(T)$ is the *Kleisli category* of the monad T, which may be viewed as the subcategory of free T-algebras within the Eilenberg-Moore category of T. This equivalence was first asserted by F. E. J. Linton in [7], although in a more general way. In this work, we will give a self-contained proof that only involves basic concepts of category theory here defined.

Monads are pervasive structures in many mathematical fields, and furthermore, they are a fundamental concept of *functional programming*. Functional programming is a programming paradigm that treats computation as the mathematical

evaluation of functions, avoiding mutable data. Because of that, functional programming is very sensible to side effects and exceptions. A *monad* in functional programming is viewed as an amplifier of types that give extra structure so that some properties are met, allowing side effects to be controlled without the need to write unnecessary code (see Section 3 for a detailed example). Monads are the base and core of the programming language Haskell, which is used in many different applications, such as the spam filter of Facebook.

Back to universal algebra, another feature of its relation with category theory is that the equivalence between the category of Lawvere theories and the category of finitary monads on sets goes through models of theories and algebras over monads. Indeed, we will prove that, given a Lawvere theory L, there is an equivalence between the category of models of L in Set and the category of algebras over the associated monad T_L. Specifically, to a T_L algebra (A, a) we associate the only model $M\colon \mathcal{L} \to \mathsf{Set}$ such that

$$M(1) = A \text{ and } M\varphi = a_n(\varphi, -) \text{ for each } \varphi \in \mathcal{L}(n, 1),$$

where a_n denotes the component of a acting on $\mathcal{L}(n, 1) \times A^n$ (see Section 4). Conversely, for a model $M\colon \mathcal{L} \to \mathsf{Set}$, we consider the T_L-algebra

$$(M(1),\, U\varepsilon_M\colon T_L M(1) \to M(1))$$

where $\varepsilon_M\colon F_L U M \to M$ is the counit of the adjoint pair (F_L, U) described above.

This equivalence prompts the possibility to translate results involving algebras over monads to results about models of Lawvere theories. We aim to explore, in subsequent work, possible applications of this fact, based on results obtained in [3]. As a first instance, we give here a new proof of the fact that every localization on the category of abelian groups lifts (in a unique way) to a localization on R-modules for every ring R with 1.

2 Preliminaries

Category theory formalizes mathematical structures in terms of directed graphs called *categories*, whose nodes are called *objects* and whose directed edges are called *morphisms*. The concept was first introduced by S. Eilenberg and S. Mac Lane in 1942. They were interested in the understanding of the processes that preserve mathematical structures, particularly in the field of algebraic topology. The first definition they gave was a purely abstract definition of a category along the lines of the axiomatic definition of a group. This definition evolved over time, depending on set-theoretical foundations and on the specific goals intended in each context. Here we present what is nowadays a standard definition in category theory.

Definition 2.1. A *category* $\mathcal{C}$ consists of

(i) a collection $\mathrm{Ob}(\mathcal{C})$ of *objects*;

(ii) for all $C, D \in \mathrm{Ob}(\mathcal{C})$, a set $\mathcal{C}(C, D)$ of *morphisms from C to D*;

(iii) for all $C, D, E \in \mathrm{Ob}(\mathcal{C})$, a function

$$\mathcal{C}(D, E) \times \mathcal{C}(C, D) \longrightarrow \mathcal{C}(C, E)$$

called *composition* and denoted $(g, f) \mapsto g \circ f$;

(iv) for each $C \in \mathrm{Ob}(\mathcal{C})$, an element id_C of $\mathcal{C}(C, C)$, called the *identity on C*, satisfying the following axioms:

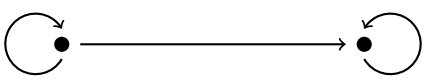
- associativity: for all $f \in \mathcal{C}(C, D)$, $g \in \mathcal{C}(D, E)$, and $h \in \mathcal{C}(E, F)$, we have

$$(h \circ g) \circ f = h \circ (g \circ f);$$

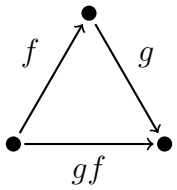
- identity laws: for each $f \in \mathcal{C}(C, D)$ we have $f \circ \mathrm{id}_C = f = \mathrm{id}_D \circ f$.

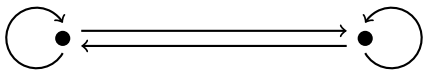
With this simple and abstract definition, most mathematical structures can be viewed as categories. However, this definition covers more than ordinary mathematical tools. A category can be specified by saying directly what its objects, morphisms, composition and identities are. For example, there is a category $\emptyset$ with no objects. There is also a category $\mathbf{1}$ with exactly one object and the identity morphism of this object.

There is another category with two objects (with their identities) and one morphism between them —composition is defined in the only possible way.



This idea leads to more complicated examples, all of which can be drawn like directed graphs.





These examples make clear that *morphisms* need not be the usual maps found in algebraic structures. They can be defined in a purely formal way, just by stating how they combine with other morphisms. Examples also show that, contrary to what one could expect of such a definition, categories need not be enormous. Some are small, manageable structures that one can completely specify.

Invertible morphisms are called *isomorphisms*. These can be viewed as pairs of opposite orientations in a given edge of a graph whose two composites are identities.

Example 2.2. There is a category Set whose objects are sets and whose morphisms are ordinary functions between them. Composition and identities are those of functions.

Example 2.3. There is a category Grp whose objects are groups and whose morphisms are group homomorphisms.

Example 2.4. Similarly, there is a category Ab of abelian groups with group homomorphisms; a category Ring of rings with ring homomorphisms; a category Vect_k of vector spaces over a field k with linear maps; a category Top of topological spaces with continuous maps, and many more.

As mentioned before, Eilenberg and Mac Lane were interested in the processes that preserve mathematical structures. For this purpose, they defined the concept of a *functor*. Similarly as in the previous examples, a functor is a "map" between categories.

Definition 2.5. Let $\mathcal{C}$ and $\mathcal{D}$ be categories. A *functor* $F\colon \mathcal{C} \to \mathcal{D}$ consists of a function $\mathrm{Ob}(\mathcal{C}) \to \mathrm{Ob}(\mathcal{D})$ written as $C \mapsto FC$, and, for all $C, C' \in \mathrm{Ob}(\mathcal{C})$, a function

$$\mathcal{C}(C, C') \longrightarrow \mathcal{D}(F(C), F(C'))$$

written as $f \mapsto Ff$, satisfying the following conditions:

- $F(f' \circ f) = Ff' \circ Ff$ whenever f and f' are composable in $\mathcal{C}$;
- $F(\mathrm{id}_C) = \mathrm{id}_{FC}$ for every $C \in \mathrm{Ob}(\mathcal{C})$.

Example 2.6. There are many functors between categories, but perhaps one of the most common kinds of functors are the so-called *forgetful* functors. For instance, consider the functor $U\colon \mathsf{Grp} \to \mathsf{Set}$ that, given a group G, returns the underlying (hence the choice of U as a name) set of G (that is, the set of its elements). If $f\colon G \to H$ is a group homomorphism, then Uf is the function f itself, just *forgetting* that G and H are groups. The general fact is that U forgets additional structure.

Example 2.7. Similarly, there are forgetful functors $U\colon \mathsf{Ring} \to \mathsf{Set}$ forgetting the ring structure; $U\colon \mathsf{Ring} \to \mathsf{Ab}$ that forgets the multiplicative structure and returns the underlying additive group; $U\colon \mathsf{Ab} \to \mathsf{Grp}$ forgetting commutativity; and $U\colon \mathsf{Top} \to \mathsf{Set}$ forgetting the topology and returning the set of points.

Example 2.8. There is a dual concept of the forgetful functors: those that *add* some kind of additional structure, called *free* functors. Given a set A, one can build the *free group over* A, thus yielding a functor $F\colon \mathsf{Set} \to \mathsf{Grp}$. For each set A, the group FA is obtained from A by adding to it sufficiently many elements until it becomes a group, but without imposing any equations other than those required by the definition of a group.

More precisely, the elements of FA are formal expressions called *words*, formed by elements of A and their formal inverses. Multiplication is concatenation of words; two words are equal if one can be obtained from the other by cancellation of elements adjacent to their inverses. Every function $f\colon A \to A'$ gives rise to a group homomorphism $Ff\colon FA \to FA'$.

Example 2.9. Similarly as in the previous example, there are free functors such as the free abelian group functor $F\colon \mathsf{Set} \to \mathsf{Ab}$ or the abelianization functor $F\colon \mathsf{Grp} \to \mathsf{Ab}$. The free functor $F\colon \mathsf{Set} \to \mathsf{Top}$ endows every set with the discrete topology.

Example 2.10. There is also a free functor $F\colon \mathsf{Ring} \to \mathsf{Ring}_1$ from the category of rings to the category of rings with 1. Given a ring R, this functor adds to R a new element that acts as an identity for multiplication. This is similar to the free functor $F\colon \mathsf{Top} \to \mathsf{Top}_*$ to the category of pointed topological spaces that adds to every space a disjoint basepoint.

Functors are morphisms between categories and they are useful in that they relate a category with other categories. Since categories are intended to model all kinds of mathematical structures, one may be wondering if it is possible to treat functors as categories themselves. Indeed, we can move one step forward and consider transformations between functors. Such transformations provide a way to convert one functor into another in terms of its internal structure (identities and composition). This is reminiscent to homotopies in algebraic topology, which are also "maps between maps".

Definition 2.11. Let $\mathcal{C}$ and $\mathcal{D}$ be categories and let $F\colon \mathcal{C} \to \mathcal{D}$ and $G\colon \mathcal{C} \to \mathcal{D}$ be functors. A *natural transformation* $\alpha\colon F \to G$ is a family $(\alpha_C\colon FC \to GC)_C$ of morphisms in $\mathcal{D}$, one for each object $C \in \mathrm{Ob}(\mathcal{C})$, such that, for every morphism $f\colon C \to C'$ in $\mathcal{C}$, the square

$$\begin{array}{ccc} FC & \xrightarrow{Ff} & FC' \\ {\scriptstyle \alpha_C}\downarrow & & \downarrow{\scriptstyle \alpha_{C'}} \\ GC & \xrightarrow{Gf} & GC' \end{array}$$

commutes. The morphisms α_C are called the *components* of α.

Example 2.12. Let $\mathcal{C}$ be a discrete category (that is, one in which the only morphisms are identities), and let $F, G\colon \mathcal{C} \to \mathcal{D}$ be functors, where $\mathcal{D}$ is any category. Then F and G are just families $(FC)_C$ and $(GC)_C$ of objects in $\mathcal{D}$. A natural transformation $\alpha\colon F \to G$ is a family $(\alpha_C\colon FC \to GC)_C$ of morphisms in $\mathcal{D}$.

If two functors $F\colon \mathcal{C} \to \mathcal{D}$ and $G\colon \mathcal{D} \to \mathcal{C}$ are such that $G \circ F = \mathrm{Id}_{\mathcal{C}}$ and $F \circ G = \mathrm{Id}_{\mathcal{D}}$, we say that $\mathcal{C}$ and $\mathcal{D}$ are *isomorphic* and that F and G are inverse *isomorphisms* of categories.

However, it is rather infrequent to find isomorphisms between categories. The degree of abstraction in category theory implies that precise equalities are often useless. For this purpose, we are interested in the concept of *equivalence* of categories, which we next define, rather than isomorphism.

Definition 2.13. A *natural isomorphism* of functors is a natural transformation of functors whose components are isomorphisms.

Thanks to this definition, we can introduce the concept of equivalence of categories. This concept establishes whether two categories are "essentially the same". Establishing an equivalence involves demonstrating strong similarities between the mathematical structures concerned. Sometimes, those structures may seem totally unrelated at a superficial level. In those cases, equivalences are especially powerful: they create the opportunity to "translate" theorems between such mathematical structures, knowing that the essential meaning of those theorems is preserved under the translation. In some way, this is the main goal of this work: to establish a relation between monads and Lawvere theories that allows us to translate certain theorems.

Definition 2.14. An *equivalence* between categories $\mathcal{C}$ and $\mathcal{D}$ consists of a pair of functors $F\colon \mathcal{C} \to \mathcal{D}$ and $G\colon \mathcal{D} \to \mathcal{C}$ together with natural isomorphisms

$$\eta\colon \mathrm{Id}_{\mathcal{C}} \to G \circ F, \qquad \varepsilon\colon F \circ G \to \mathrm{Id}_{\mathcal{D}}.$$

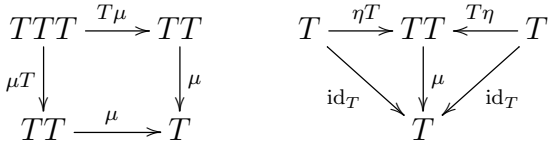
If there exists such an equivalence, we say that $\mathcal{C}$ and $\mathcal{D}$ are *equivalent*, and write $\mathcal{C} \simeq \mathcal{D}$. We also say that the functors F and G are inverse *equivalences* of categories.

3 Monads and adjunctions

The concept of a *monad* was first introduced by R. Godement in 1958 (although, more precisely, he discussed *comonads* in [5]). Godement was studying sheaf theory, which is a way to capture local data about a manifold and, in doing so, obtaining global properties. His definition was first named *standard construction*, which is entirely obsolete, or *triple*, which is still used nowadays. The concept was later renamed "monad" by Mac Lane, because of the analogy with *monoids*. Monads are frequent in many branches of mathematics: from universal algebra, which is the ultimate goal of this essay —focusing on Lawvere theories— to mathematical analysis (since Cauchy completions are monads). Monads also gave birth to a new mindset for theoretical computer science.

3.1 Definition and examples

Definition 3.1. A *monad* on a category $\mathcal{C}$ is a functor $T\colon \mathcal{C} \to \mathcal{C}$ equipped with natural transformations $\eta\colon \mathrm{Id}_{\mathcal{C}} \to T$ (the *unit*) and $\mu\colon TT \to T$ (the *multiplication*) such that the following diagrams commute in $\mathcal{C}$:



Example 3.2. The powerset functor $P\colon \mathsf{Set} \to \mathsf{Set}$ is a monad on the category Set of sets. It maps a set to the set of its subsets and, given a function $f\colon A \to B$ between sets, Pf maps a set to its image under f. The unit $\eta_A\colon A \to PA$ sends an element to the singleton subset; the multiplication $\mu_A\colon PPA \to PA$ takes the union of a set of subsets.

Example 3.3. The **Giry monad** acts on the category Meas of measurable spaces and functions between them. It sends a measurable space A to the measurable space $\mathrm{Prob}(A)$ of probability measures on A. The unit is the measurable function $\eta_A\colon A \to \mathrm{Prob}(A)$ that sends each element $a \in A$ to the Dirac measure (which assigns a subset the probability 1 if it contains a or 0 otherwise). The multiplication is defined using integration; details are given in [4].

Example 3.4. For every ring R with 1 there is a monad $T\colon \mathsf{Ab} \to \mathsf{Ab}$ on the category Ab of abelian groups that sends each abelian group A to the tensor product $R \otimes A$ equipped with the natural morphism $\eta_A\colon A \to R \otimes A$ sending $a \mapsto 1 \otimes a$ for all a. The multiplication $\mu_A\colon TTA \to TA$ comes from the multiplication in the ring R.

Example 3.5. There is a monad $T\colon \mathsf{Ab} \to \mathsf{Ab}$ that sends each abelian group A to the quotient $A/\tau A$ where τA is the torsion subgroup of A. In other words, T *kills* the torsion from A. This monad is *idempotent*, in the sense that $\mu_A\colon TTA \to TA$ is a natural isomorphism for all A.

3.2 From adjunctions to monads

It is natural to ask oneself the properties that should be met for a functor to be part of a monad. For this purpose, let us retake a pair of special functors: *free* and *forgetful*. In our examples, those functors seem to act "inversely". Although this notion is not entirely true, it captures the idea of *adjunctions*. Let us see, in detail, how this process works.

Denote by Vec_k the category of vector spaces over a field k and consider the fields $\mathbb{R}$ and $\mathbb{C}$. There is a forgetful functor $U\colon \mathsf{Vec}_{\mathbb{C}} \to \mathsf{Vec}_{\mathbb{R}}$. Given a vector space $E_{\mathbb{C}}$ over $\mathbb{C}$, we let $UE_{\mathbb{C}}$ be the same set as $E_{\mathbb{C}}$ but viewed as a vector space over $\mathbb{R}$ by restriction of scalars (that is, treating a real number as a complex number with zero imaginary part).

Given a $\mathbb{C}$-linear map $f\colon E_{\mathbb{C}} \to W_{\mathbb{C}}$, we can think of Uf as an $\mathbb{R}$-linear map

$$Uf\colon UE_{\mathbb{C}} \longrightarrow UW_{\mathbb{C}}$$

by defining $(Uf)(v) = f(v)$ for all $v \in UE_{\mathbb{C}}$. Moreover, U is a *faithful* functor, which means that the following function is injective:

$$\mathsf{Vec}_{\mathbb{C}}(E_{\mathbb{C}}, W_{\mathbb{C}}) \longrightarrow \mathsf{Vec}_{\mathbb{R}}(UE_{\mathbb{C}}, UW_{\mathbb{C}}).$$

Now we would like to find a free functor $F\colon \mathsf{Vec}_{\mathbb{R}} \to \mathsf{Vec}_{\mathbb{C}}$, as in Examples 2.6 and 2.8. Such a functor does not always exist and, even if it exists, it may be difficult to describe. This process often requires other mathematical tools than just forgetting additional structure. However, in this case, consider $FE_{\mathbb{R}} = \mathbb{C} \otimes E_{\mathbb{R}}$. This is the quotient $(\mathbb{C} \times E_{\mathbb{R}})/\!\sim$, where the relation $\sim$ imposes the usual bilinear conditions over $\mathbb{R}$.

Let us work a little bit more with this example to understand what is really happening there. The elements of $\mathbb{C} \otimes E_{\mathbb{R}}$ can be written as

$$z_1 \otimes v_1 + z_2 \otimes v_2 + \cdots + z_n \otimes v_n,$$

where $z_k \in \mathbb{C}$ and $v_k \in E_{\mathbb{R}}$. Since $z_k = x_k + iy_k$ for all k, the previous expression can be rewritten as

$$1 \otimes w_1 + i \otimes w_2$$

for some $w_1, w_2 \in E_{\mathbb{R}}$. Therefore, every element of $\mathbb{C} \otimes E_{\mathbb{R}}$ can be written as $1 \otimes w_1 + i \otimes w_2$.

Given an $\mathbb{R}$-linear map $f\colon E_{\mathbb{R}} \to W_{\mathbb{R}}$, we can define its $\mathbb{C}$-linear counterpart

$$Ff\colon FE_{\mathbb{R}} \longrightarrow FW_{\mathbb{R}}$$

as $(Ff)(z \otimes v) = z \otimes f(v)$. This functor F is called a *complexification* functor.

There is a map $\eta\colon E_{\mathbb{R}} \to UFE_{\mathbb{R}}$ defined as

$$\eta(v) = 1 \otimes v,$$

which is $\mathbb{R}$-linear, since $\eta(\lambda v) = 1 \otimes (\lambda v) = \lambda \otimes v = \lambda(1 \otimes v) = \lambda\eta(v)$ for all $v \in E_{\mathbb{R}}$ and $\lambda \in \mathbb{R}$.

There is also a map $\varepsilon\colon FUW_{\mathbb{C}} \to W_{\mathbb{C}}$ defined by

$$\varepsilon(z \otimes v) = zv,$$

which is $\mathbb{C}$-linear since $\varepsilon(z'(z \otimes v)) = \varepsilon(z'z \otimes v) = (z'z)v = z'(zv) = z'\varepsilon(z \otimes v)$.

Hence, we can define a map

$$\Phi\colon \mathsf{Vec}_{\mathbb{R}}(E_{\mathbb{R}}, UW_{\mathbb{C}}) \longrightarrow \mathsf{Vec}_{\mathbb{C}}(FE_{\mathbb{R}}, W_{\mathbb{C}})$$

as $\Phi(f)(z \otimes v) = zf(v) = \varepsilon(z \otimes f(v))$, which is bijective and has an inverse

$$\Psi\colon \mathsf{Vec}_{\mathbb{C}}(FE_{\mathbb{R}}, W_{\mathbb{C}}) \longrightarrow \mathsf{Vec}_{\mathbb{R}}(E_{\mathbb{R}}, UW_{\mathbb{C}})$$

defined by $\Psi(g)(v) = g(1 \otimes v) = g(\eta(v))$. Therefore, Φ maps the identity of $UW_{\mathbb{C}}$ to the map ε and, similarly, Ψ maps the identity of $FE_{\mathbb{R}}$ to η.

Note that $UFE_{\mathbb{R}}$ is again a vector space over $\mathbb{R}$, of twice the dimension of $E_{\mathbb{R}}$, equipped with a natural inclusion $E_{\mathbb{R}} \to UFE_{\mathbb{R}}$. Moreover, we can iterate UF and consider $UFUFE_{\mathbb{R}}$, for which there is a natural map into $UFE_{\mathbb{R}}$ coming from the multiplication $\mathbb{C} \otimes \mathbb{C} \to \mathbb{C}$ of complex numbers. These are precisely the ingredients of a monad.

This process is describing that, although the functors U and F are not inverse of each other, the maps $FE_{\mathbb{R}} \to W_{\mathbb{C}}$ and $E_{\mathbb{R}} \to UW_{\mathbb{C}}$ are closely related. This kind of relation is formalized by means of the following definition.

Definition 3.6. Let $F\colon \mathcal{C} \rightleftarrows \mathcal{D} : G$ be two categories and two functors. We say that F is *left adjoint* to G and G is *right adjoint* to F if

$$\mathcal{D}(FC, D) \cong \mathcal{C}(C, GD)$$

naturally in $C \in \mathrm{Ob}(C)$ and $D \in \mathrm{Ob}(\mathcal{D})$.

Example 3.7. Let $\mathcal{P}$ be a preordered set (that is, a set equipped with a relation $\leq$ that is reflexive and transitive). Then $\mathcal{P}$ can be considered as a category whose objects are the elements of $\mathcal{P}$ and for which there is an unique morphism $x \to y$ whenever $x \leq y$. In this case, $\mathcal{P}$ is called a *preorder category*.

We define the concept of *Galois connection* (f, g) between two preordered categories $\mathcal{P}$ and $\mathcal{Q}$ as a pair of order-preserving maps

$$f\colon \mathrm{Ob}\,\mathcal{P} \rightleftarrows \mathrm{Ob}\,\mathcal{Q} : g$$

such that $f(x) \leq y$ if and only if $x \leq g(y)$, for $x \in \mathrm{Ob}\,\mathcal{P}$ and $y \in \mathrm{Ob}\,\mathcal{Q}$.

Each Galois connection (f, g) gives rise to a pair of adjoint functors between $\mathcal{P}$ and $\mathcal{Q}$. Let us see a specific example. Consider $\mathcal{P} = (\mathbb{Z}, \leq)$ and $\mathcal{Q} = (\mathbb{R}, \leq)$ with their usual orders. Then the functions

$$\iota\colon \mathbb{Z} \rightleftarrows \mathbb{R} : \lfloor \cdot \rfloor$$

where ι represents the inclusion and $\lfloor \cdot \rfloor$ the floor function (returning the integral part of every real number), yield an adjoint pair since

$$\iota(z) \leq r \text{ if and only if } z \leq \lfloor r \rfloor \text{ for all } z \in \mathbb{Z},\ r \in \mathbb{R}.$$

Proposition 3.8. *Given two categories $\mathcal{C}$ and $\mathcal{D}$, every pair of adjoint functors $F\colon \mathcal{C} \leftrightarrows \mathcal{D} : U$ give rise to a monad $T = UF$ on $\mathcal{C}$.*

Proof. When looking at books and articles about category theory, no matter how basic they are, this proof is always omitted or considered routine. Because of this, we will present here an extensive, perhaps too detailed, proof.

Since F and U are adjoint functors, there exists a natural isomorphism $\Phi_{X,Y}$ such that $\mathcal{D}(FX, Y) \cong \mathcal{C}(X, UY)$. By definition, $\eta_X = \Phi_{X,FX}(\mathrm{id}_{FX})$ (the unit) and $\varepsilon_Y = (\Phi_{UY,Y})^{-1}(\mathrm{id}_{UY})$ (the counit). Since Φ is a natural isomorphism, $\eta\colon \mathrm{Id}_{\mathcal{C}} \to UF$ and $\eta\colon FU \to \mathrm{Id}_{\mathcal{D}}$ are natural transformations.

First, let us prove that $Uf \circ \eta_X = \Phi_{X,Y}(f)$ for all $f\colon FX \to Y$. Due to naturality, the following diagram commutes:

$$\begin{array}{ccc} \mathcal{D}(FX, Y) & \xrightarrow{\Phi_{X,Y}} & \mathcal{C}(X, UY) \\ {\scriptstyle f_*}\uparrow & & \uparrow{\scriptstyle (Uf)_*} \\ \mathcal{D}(FX, FX) & \xrightarrow{\Phi_{X,FX}} & \mathcal{C}(X, UFX) \end{array}$$

That is to say,

$$(Uf)_* \circ \Phi_{X,FX} = \Phi_{X,Y} \circ f_*$$

or, equivalently,

$$(Uf)_*(\Phi_{X,FX}(\mathrm{id}_{FX})) = \Phi_{X,Y}(f_*(\mathrm{id}_{FX})).$$

By definition

$$(Uf)_*(\eta_X) = \Phi_{X,Y}(f)$$

and finally

$$Uf \circ \eta_X = \Phi_{X,Y}(f).$$

Similarly, $\varepsilon_Y \circ Fg = (\Phi_{X,Y}(g))^{-1}$ for all $g\colon X \to UY$.

Now, let us consider $UF\colon \mathcal{C} \to \mathcal{C}$ and write $T = UF$. We will prove that T is a monad, by letting $\eta\colon \mathrm{Id}_{\mathcal{C}} \to T$ be the unit of the adjunction and $\mu\colon TT \to T$ be defined as $\mu_X = U\varepsilon_{FX}$.

We have a diagram

$$\begin{array}{ccccc} T & \xrightarrow{\eta_T} & TT & \xleftarrow{T\eta} & T \\ & {\scriptstyle \mathrm{id}_T}\searrow & \downarrow{\scriptstyle \mu} & \swarrow{\scriptstyle \mathrm{id}_T} & \\ & & T & & \end{array}$$

which can be rewritten as

$$\begin{array}{ccccc} UFX & \xrightarrow{\eta_{UFX}} & UFUFX & \xleftarrow{UF\eta_X} & UFX \\ & {\scriptstyle \mathrm{id}_{UFX}}\searrow & \downarrow{\scriptstyle \mu_X} & \swarrow{\scriptstyle \mathrm{id}_{UFX}} & \\ & & UFX & & \end{array}$$

Let us check that it commutes:

$$\mu_X \circ \eta_{UFX} = U\varepsilon_{FX} \circ \eta_{UFX} = \Phi_{UFX,FX}(\varepsilon_{FX}) = \mathrm{id}_{UFX}$$

and

$$\mu_X \circ UF\eta_X = U(\varepsilon_{FX} \circ F\eta X) = U((\Phi_{X,FX})^{-1}(\eta_X)) = U(\mathrm{id}_{FX}) = \mathrm{id}_{UFX}.$$

We must check that the following diagram commutes too:

$$\begin{array}{ccc} TTT & \xrightarrow{T\mu} & TT \\ {\scriptstyle \mu T}\downarrow & & \downarrow{\scriptstyle \mu} \\ TT & \xrightarrow{\mu} & T \end{array}$$

It can be rewritten as

$$\begin{array}{ccc} UFUFUFX & \xrightarrow{UF\mu_X} & UFUFX \\ {\scriptstyle \mu_{UFX}}\downarrow & & \downarrow{\scriptstyle \mu_X} \\ UFUFX & \xrightarrow{\mu_X} & UFX \end{array}$$

so we must prove that $\mu_X \circ UF\mu_X = \mu_X \circ \mu_{UFX}$ for all $X \in \mathcal{C}$. Since $\mu_X = U\varepsilon_{FX}$, this is just

$$U\varepsilon_{FX} \circ UFU\varepsilon_{FX} = U\varepsilon FX \circ U\varepsilon_{FUFX}$$

and, operating with U, we get

$$U(\varepsilon_{FX} \circ FU\varepsilon_{FX}) = U(\varepsilon_{FX} \circ \varepsilon_{FUFX}).$$

Since U is a functor, it will be enough to prove that

$$\varepsilon_{FX} \circ FU\varepsilon_{FX} = \varepsilon_{FX} \circ \varepsilon_{FUFX}$$

or, equivalently,

$$\varepsilon_{FX} \circ FU\varepsilon_{FX} = (\Phi_{UFUFX,FX})^{-1}(U\varepsilon_{FX})$$
$$\Phi_{UFUF,FX}(\varepsilon_{FX} \circ FU\varepsilon_{FX}) = U\varepsilon_{FX}.$$

Since Φ is a bijection for every pair of objects, it is enough to prove that

$$\Phi_{UFUFX,FX}(\varepsilon_{FX} \circ \varepsilon_{FUFX}) = U\varepsilon_{FX}.$$

Finally, operating, we have that

$$\Phi_{UFUFX,FX}(\varepsilon_{FX} \circ \varepsilon_{FUFX}) = U(\varepsilon_{FX} \circ \varepsilon_{FUFX}) \circ \eta_{UFUFX} =$$
$$= U\varepsilon_{FX} \circ (U\varepsilon_{FUFX} \circ \eta_{UFUFX}) = U\varepsilon_{FX} \circ \Phi_{UFUFX,FUFX}(\varepsilon_{FUFX}) =$$
$$= U\varepsilon_{FX} \circ \mathrm{id}_{UFUFX} = U\varepsilon_{FX}.$$

□

The relationship between adjunctions and monads yields a wide variety of examples from every branch of Mathematics.

Example 3.9. There is a forgetful functor $U\colon \mathsf{Cat} \to \mathsf{DirGraph}$ that admits a left adjoint F, defining the *free category on a directed graph*. A directed graph G consists of a set V of vertices, a set E of edges and two functions $s, t\colon E \rightrightarrows V$ defining the source and target of each directed edge. The free category on G has V as its set of objects and identities for each vertex together with finite paths of edges as morphisms. Composition is defined by concatenation of paths.

The adjunction induces a monad on $\mathsf{DirGraph}$ that carries a directed graph G to the graph with the same vertices but whose edges are finite directed paths of edges in G. This is the underlying directed graph of the free category G.

Example 3.10. The free-forgetful adjunction between sets and groups induces the *free group monad* $F\colon \mathsf{Set} \to \mathsf{Set}$ that sends a set A to the set FA of finite words in letters $a \in A$ together with formal inverses a^{-1}.

Example 3.11. There are functors $D\colon \mathsf{Set} \to \mathsf{Top}$, that equips a set with the discrete topology; $U\colon \mathsf{Top} \to \mathsf{Set}$ that sends a topological space to its set of points; and $I\colon \mathsf{Set} \to \mathsf{Top}$ that equips a set with the indiscrete topology. We have that D and U are a pair of adjoint functions, but so are U and I, meaning that we have two different monads. We have a monad over Set, arisen from the adjunction $D \dashv U$, which is the identity. We also have a monad over Top, arisen from $U \dashv I$ that sends a topological space to the space with the same set of points equipped with the indiscrete topology.

Example 3.12. The inclusion functor of the category of complete metric spaces with uniformly continuous mappings to the category of metric spaces has a left adjoint. Such adjoint is the completion of a metric space on objects and the extension by density on arrows. This yields a monad on metric spaces.

Example 3.13. There is a monad $T\colon \mathsf{Ab}_* \to \mathsf{Ab}_*$ on the category Ab_* of abelian groups with a distinguished element that, given an abelian group A and an element $e \in A$, returns the underlying abelian group of a free ring on A with unit e. This may sound surprising, but it comes from a rather simple adjunction. Let us denote by Ring_1 the category of rings with unit. There is a functor $U\colon \mathsf{Ring}_1 \to \mathsf{Ab}_*$ that forgets the ring structure and returns the underlying abelian group. As in most adjunctions, the left adjoint $F\colon \mathsf{Ab}_* \to \mathsf{Ring}_1$ is more complicated. To understand it, let us look at rings with another mindset.

Suppose that A is a ring with unit. Then the multiplication on A yields a bilinear map

$$\mu\colon A \times A \longrightarrow A,$$

where we write $\mu(a, b) = a \cdot b$ and require μ to be associative. This map μ can also be seen as a homomorphism of abelian groups

$$\begin{array}{rccc} \mu: & A \otimes A & \longrightarrow & A \\ & a \otimes b & \longmapsto & a \cdot b \end{array}$$

which is associative. Distributivity follows from the properties of the tensor product. There is also a group homomorphism

$$\begin{array}{rccc} \eta: & \mathbb{Z} & \longrightarrow & A \\ & 1 & \longmapsto & e \end{array}$$

that provides A with a unit for μ, i.e., $\mu(a,e) = a = \mu(e,a)$ for all $a \in A$. Thus, the group A equipped with the functions μ and η is a monoid in the category of abelian groups, and this is precisely a ring with unit.

Now, given an abelian group A with a distinguished element e, how can we turn it into a monoid? The answer is hidden in the properties of the tensor product. Consider

$$F(A,e) = A \oplus (A \otimes A) \oplus (A \otimes A \otimes A) \oplus \ldots / \sim$$

where $\sim$ denotes an equivalence relation such that $a \sim e \otimes a \sim a \otimes e$ for all $a \in A$, $e \otimes a \otimes b \sim a \otimes e \otimes b \sim a \otimes b \otimes e \sim a \otimes b$ for all $a, b \in A$, and so on. The properties of the direct sum and those of the tensor product ensure that $F(A,e)$ is a ring with unit e.

Note that the equivalence relation $\sim$ would not be necessary if we were not asking the ring to have a prescribed unit. Similarly, further conditions can be imposed within the equivalence relation if we want to obtain additional properties on the ring structure, such as commutativity.

Example 3.14. The previous example is a good algebraic introduction to the **James construction**. The James construction is a monad $J\colon \mathsf{Top}_* \to \mathsf{Top}_*$ on the category Top_* of topological spaces with a basepoint, coming from an adjunction

$$\mathsf{Top}_* \underset{U}{\overset{F}{\rightleftarrows}} \mathsf{MonTop}$$

where MonTop stands for topological monoids (with the unit as basepoint). While U denotes the forgetful functor as usual, the functor F may be defined in the same fashion as in the previous example. Thus, consider a topological space X with a basepoint x_0 and define

$$F(X,x_0) = X \vee (X \times X) \vee (X \times X \times X) \vee \ldots / \sim$$

where $\sim$ is an equivalence relation used to impose that x_0 acts as a unit, namely

$$x \sim (x_0, x) \sim (x, x_0)$$

for all $x \in X$. The points of X can be multiplied in $F(X,x_0)$ by defining $x \cdot y = (x,y)$ for $x, y \in X$, and in fact $F(x,x_0)$ becomes a monoid by extending this multiplication over all summands. The fact that $F(X,x_0)$ is a free topological monoid over X follows from the properties of the wedge sum and those of the Cartesian product.

There are some topological spaces, such as the circle S^1 or the 3-sphere S^3, where a multiplication is already defined. Those are topological monoids. However,

there are many other spaces where no multiplication exists. Thanks to the James construction, it becomes possible to take two points of any space and multiply them formally.

The usual notation for the James construction on a space X is JX. As in the algebraic counterpart, the equivalence relation defining JX may be replaced by a stronger one so as to require additional properties for a topological monoid (such as commutativity).

Example 3.15. There is a functor $U\colon \mathsf{Set}_+ \to \mathsf{Set}$ from pointed sets to sets that forgets the basepoint; this functor has a left adjoint $F\colon \mathsf{Set} \to \mathsf{Set}_+$ that sends a set A to the pointed set $A_+ = A \bigsqcup \{a\}$ with a disjoint basepoint. This adjunction induces a monad $T_+\colon \mathsf{Set} \to \mathsf{Set}$ that adds a new point at a time. The components of the unit are given by the obvious inclusion $\eta_A\colon A \to A_+$. The components of the multiplication $\mu_A\colon (A_+)_+ \to A_+$ are defined to be the identity on the subset A and to send the two new points in $(A_+)_+$ to the new point in A_+. In Computer Science, this is called the **maybe monad**. It is further discussed in Section 3.4.

3.3 From monads to adjunctions

After having studied several examples, it seems natural that monads arise from adjunctions. But is it possible to retrieve an adjunction from a given monad? The answer is yes: it is possible, and in multiple ways. In order to understand how, first we need to introduce more notation.

Definition 3.16. Let $T = (T, \mu, \eta)$ be a monad on a category $\mathcal{C}$. A *T-algebra* in $\mathcal{C}$ is a pair (A, a) of an object A in $\mathcal{C}$ and a morphism $a\colon TA \to A$ such that the following diagrams commute:

$$\begin{array}{ccc} TTA & \xrightarrow{\mu_A} & TA \\ {\scriptstyle Ta}\downarrow & & \downarrow{\scriptstyle a} \\ TA & \xrightarrow{a} & A \end{array} \qquad \begin{array}{ccc} A & \xrightarrow{\eta_A} & TA \\ & {\scriptstyle \mathrm{id}_A}\searrow & \downarrow{\scriptstyle a} \\ & & A \end{array}$$

At this point, it should be noted that such definitions describe structure and properties. But, as a piece of advice, finding the algebras for a monad is not always easy. Sometimes, one has to settle for the abstract definition. In other cases, one may find a way to identify them to some known structure. In some other cases, the known structure can be extremely difficult.

Example 3.17. The algebras over the James construction are the topological monoids.

Example 3.18. Let us look at the example of the maybe monad. An algebra is a set A together with a map $a\colon A_+ \to A$ so that the following diagrams commute:

$$\begin{array}{ccc} (A_+)_+ & \xrightarrow{\mu_A} & A_+ \\ {\scriptstyle Ta}\downarrow & & \downarrow{\scriptstyle a} \\ A_+ & \xrightarrow{a} & A \end{array} \qquad \begin{array}{ccc} A & \xrightarrow{\eta_A} & A_+ \\ & {\scriptstyle \mathrm{id}_A}\searrow & \downarrow{\scriptstyle a} \\ & & A \end{array}$$

While the first one does not add additional conditions, the triangle asserts that the map $a\colon A_+ \to A$ restricts to the identity on the A component (remember that η_A was the trivial inclusion). Thus, the data of an algebra is a set with a specified basepoint: the image of the extra point under the map a.

There are two canonical (typically distinct) ways to recover an adjunction.

Definition 3.19. Let $T = (T, \mu, \eta)$ be a monad on a category $\mathcal{C}$. The *Eilenberg-Moore* category of T is the category $\mathcal{C}^T$ whose objects are the T-algebras (A, a) and whose morphisms $(A, a) \to (B, b)$ are maps $f\colon A \to B$ in $\mathcal{C}$ such that the following diagram commutes in $\mathcal{C}$:

$$\begin{array}{ccc} TA & \xrightarrow{Tf} & TB \\ {\scriptstyle a}\downarrow & & \downarrow{\scriptstyle b} \\ A & \xrightarrow{f} & B \end{array}$$

Example 3.20. Recalling the pointed set monad with its algebras, a morphism $(A, a) \to (B, b)$ is a map $f\colon A \to B$ such that

$$\begin{array}{ccc} A_+ & \xrightarrow{f_+} & B_+ \\ {\scriptstyle a}\downarrow & & \downarrow{\scriptstyle b} \\ A & \xrightarrow{f} & B \end{array}$$

Since a and b are the identity on the components of A and B respectively, the map f_+ carries the extra point in A_+ to the extra point in B_+. This condition demands that the basepoint of A should go to the baisepoint of B. In conclusion, the Eilenberg-Moore category of this monad is isomorphic to Set_*.

Definition 3.21. Let $T = (T, \mu, \eta)$ be a monad on a category $\mathcal{C}$. The *Kleisli* category of T is the category $\mathcal{C}_T$ whose objects are those of $\mathcal{C}$, and a morphism from A to B in $\mathcal{C}_T$ is a morphism $A \to TB$ in $\mathcal{C}$.

P. J. Hilton was the first to conjecture that every monad arises from an adjunction. The two following solutions were provided more or less simultaneously, using different constructions, by S. Eilenberg and J. C. Moore, and H. Kleisli.

Proposition 3.22. *If $T = (T, \mu, \eta)$ is a monad on a category $\mathcal{C}$, then there is a category $\mathcal{B}$ and an adjoint pair $F\colon \mathcal{C} \to \mathcal{B}$, $U\colon \mathcal{B} \to \mathcal{C}$ such that $T = UF$.*

Proof. Let us start with the Kleisli construction. Consider $\mathcal{B} = \mathcal{C}_T$, the Kleisli category of T. Define a functor $U\colon \mathcal{C}_T \to \mathcal{C}$ by $UA = TA$. Given $f \in \mathcal{C}_T(A, B)$, define Uf to be $Tf \circ \mu_B$. Define also a functor $F\colon \mathcal{C} \to \mathcal{C}_T$ by $FA = A$, and, given $f \in \mathcal{C}(A, B)$, let Ff be the composite $\eta_A \circ Tf$. It then follows from the definitions that $\mathcal{C}(A, UB) = \mathcal{C}(A, TB) \cong \mathcal{C}_T(A, B) = \mathcal{C}_T(FA, B)$.

Now let us move on to the Eilenberg-Moore construction. Let $\mathcal{B} = \mathcal{C}^T$ be the Eilenberg-Moore category of T. The functor $U\colon \mathcal{C}^T \to \mathcal{C}$ takes a T-algebra (A, a)

and returns the object A. Given $f \in \mathcal{C}^T((A,a),(B,b))$, define $Uf = f$ by simply acting upon the objects A and B and forgetting the extra structure. Define also a functor $F\colon \mathcal{C} \to \mathcal{C}^T$ by $FA = (TA, \mu_A)$. This is indeed a T-algebra, since the following diagrams commute because they come from the square and triangle identities of a monad.

$$\begin{array}{ccc} TTTA & \xrightarrow{\mu_{TA}} & TTA \\ {\scriptstyle T\mu_A}\downarrow & & \downarrow{\scriptstyle \mu_A} \\ TTA & \xrightarrow{\mu_A} & TA \end{array} \qquad \begin{array}{ccc} TA & \xrightarrow{\eta_{TA}} & TA \\ & {\scriptstyle \mathrm{id}_{TA}}\searrow & \downarrow{\scriptstyle \mu_A} \\ & & A \end{array}$$

Given $f \in \mathcal{C}(A,B)$, defined $Ff = Tf$, which is a morphism of T-algebras since the following diagram commutes because μ is a natural transformation:

$$\begin{array}{ccc} TTA & \xrightarrow{TTf} & TTB \\ {\scriptstyle \mu_A}\downarrow & & \downarrow{\scriptstyle \mu_B} \\ TA & \xrightarrow{Tf} & TB \end{array}$$

We need to check that $\mathcal{C}(A, U(B,b)) \cong \mathcal{C}^T(FA,(B,b))$, which amounts to proving that $\mathcal{C}(A,B) \cong \mathcal{C}^T((TA,\mu_A),(B,b))$. Consider

$$\Psi\colon \mathcal{C}^T((TA,\mu_A),(B,b)) \to \mathcal{C}(A,B)$$

defined by $\Psi(h) = h \circ \eta_A$, which, as we next show, is inverse to the function

$$\Phi\colon \mathcal{C}(A,B) \to \mathcal{C}^T((TA,\mu_A),(B,b))$$

defined by $\Phi(g) = b \circ UFg = b \circ Tg$.

Let us prove that $\Phi(\Psi(h)) = h$ for every $h \in \mathcal{C}^T((TA,\mu_A),(B,b)$. First we write $\Phi(\Psi(h)) = b \circ Th \circ T\eta_A$. In the following diagram, the triangle commutes since (T,η,μ) is a monad, and the right square commutes because h is a morphism of T-algebras:

$$\begin{array}{ccccc} TA & \xrightarrow{\mathrm{id}_{TA}} & TA & \xrightarrow{h} & B \\ & {\scriptstyle T\eta_A}\searrow & \uparrow{\scriptstyle \mu_A} & & \uparrow{\scriptstyle b} \\ & & TTA & \xrightarrow[Th]{} & TB \end{array}$$

Hence, $b \circ Th \circ T\eta_A = h$, as needed.

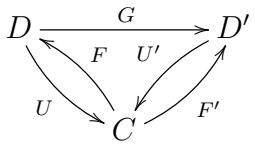
Next, we check that $\Psi(\Phi(g)) = g$ for every $g \in \mathcal{C}(A,B)$, where we view B as $U(B,b)$ for a certain T-algebra structure $b\colon TB \to B$. We have that $\Psi(\Phi(g)) = b \circ Tg \circ \eta_A$. However, $Tg \circ \eta_A = \eta_B \circ g$ since η is a natural transformation, and $b \circ \eta_B = \mathrm{id}B$ because $b\colon TB \to B$ endows B with a T-algebra structure. $\square$

These two solutions are closely related: the Kleisli category $\mathcal{C}_T$ embeds into the Eilenberg-Moore category $\mathcal{C}^T$ as the full subcategory generated by the image of the left adjoint F in the Eilenberg-Moore construction. Indeed, for all objects A and B of $\mathcal{C}$, we have

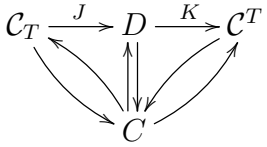
$$\mathcal{C}^T(FA,FB) \cong \mathcal{C}(A,UFB) = \mathcal{C}(A,TB) = \mathcal{C}_T(A,B).$$

This is why the Kleisli category is sometimes called the category of *free T-algebras*.

What is more, one may consider, for any monad $T = (T, \mu, \eta)$ on $\mathcal{C}$, the category Adj_T whose objects are fully specified adjunctions inducing the monad (T, μ, η). A morphism $G \in \mathsf{Adj}_T(D, D')$



is a functor commuting with both the left and right adjoints, i.e., so that $GF = F'$ and $U'G = U$. On this scenario, the Kleisli category $\mathcal{C}_T$ is initial in Adj_T and the Eilenberg-Moore category $\mathcal{C}^T$ is terminal. That is to say, for any pair of adjoint functors (U, F) between $\mathcal{C}$ and $\mathcal{D}$ inducing the monad $T = (T, \mu, \eta)$ on $\mathcal{C}$, there exist unique functors J and K



which commute with both left and right adjoints.

3.4 Monads in Computer Science

When researching about monads, examples and applications, one will notice an interesting fact: results in Computer Science (more precisely in *functional programming*) come before the mathematical ones. We know that category theory is helpful to abstract complicated results into general definitions. In other words, it is helpful within mathematics. But can it be helpful for something as tangible as Computer Science? The answer was first discussed by E. Moggi in 1991 [9].

Moggi introduced the concept of monads in the context of programming languages. In fact, his definition of a monad is exactly the same as the one we presented previously. He used monads to capture different notions of computation that go beyond total functions, such as non determinism, side effects and exceptions. Although Moggi's first approach was to study category theory in a logical way in order to understand the foundations of computation and as a *proving* tool, it has become much more.

Functional programming is a programming paradigm that treats computation as the evaluation of mathematical functions and avoids changing-state and mutable data. Haskell was born in 1990, motivated by the need of having an open standard for such functional languages. Nowadays there are many more, such as PHP, JavaScript, Perl and even some libraries to code with Python as well. What started as a theoretical approach has today very important applications, such as the spam filter in Facebook [10] and air traffic analysis tool for NATS [11]. What we mean is that category theory, and specifically monads, have real world applications through Computer Science.

In functional programming, unlike other programming paradigms, we do not have *exceptions*. This means that, if the computation goes wrong at some point, it will produce an error that will only be visible at the end of the whole computation. Exceptions allow us to stop at a set moment and decided what to do next. For example, when dividing by zero, functional programming would go down with the error while an exception could be risen and decide to skip that division. For this precise reason, monads are key for functional programming.

Monads are amplifiers of types: they add new properties to the ones we already had, so we can work around exceptions. But in order to act properly with other functions, we need to define two properties:

- the **unit** operation takes a value from a plain type and creates an equivalent monadic value. In our example, this can be seen in the inclusion $\eta_A \colon A \to A_+$, which takes a value from the set A and returns the *monadic value* in the set A_+.
- the **bind** (*multiplication*) operation that takes the value, a function that works on it and returns a new monadic value. It lets us transform operations on the unamplified type into operations on the amplified type, obeying the composition rules.

Monads are responsible for controlling side effects in functional programming, as well as acting as exceptions. They embed extra structure and logic behavior to our predefined types. So, for example, monads can provide a way to divide by zero (of course, by skipping the operation) without the need to perform further checkings in our coding. However, monads in category theory and monads in computer science are not exactly the same. The latter come from the former, but they have evolved to more practical definitions. Sometimes, it is possible to find the exact mathematical concept that gives birth to a monad. Sometimes, the computer science monad is just a reminiscence of the general definition.

As a case study, we will present a monad whose relation with the mathematical world is clear. Recall the monad from Example 3.15, whose name of **maybe monad** comes from the fact that a map in the Kleisli category is a function $A \to B_+$ which may be thought of as a partially defined function from A to B: the elements of A that are sent to the free basepoint have "undefined output".

The maybe monad is a data type that is either a single value or no value at all. It is used as a checked exception: at any point that the computation might fail, the rest of the code will be skipped and *nothing* will be returned; otherwise, it will return the proper value. Let us see an example based on `C#` code to really understand it. Suppose we have a function

```
int division_by(int x) {
        return a / x;
}
```

But we want to be able to have x null and work with that with a safe code. One possibility is to consider the operator `Nullable<T>` and reconvert our function to work properly.

```
Nullable<int> division_by(Nullable<int> x)
{
    if (x == null)
        return null;
    else
        return new Nullable<int>(a / x);
}
```

You can always have your original `int`, without the amplified value, calling the `Value` property. The problem now is how do we combine old functions that did not know about the amplified type with this one.

```
static Nullable<T> Bind<T>(Nullable<T> amplified,
Func<T, Nullable<T>> func)
{
    if (amplified == null)
        return null;
    else
        return func(amplified.Value);
}
```

This means that any method that takes an int and returns an int, or takes an int and returns a `Nullable<int>` can now have the nullable semantics applied to it. This would be a *dummy* implementation of the **maybe monad**. Out of curiosity, this would be the implementation in `Haskell`:

```
return :: a -> Maybe a
    return x  = Just x

    (>>=) :: Maybe a -> (a -> Maybe b) -> Maybe b
    (>>=) m g = case m of
                  Nothing -> Nothing
                  Just x  -> g x
```

The maybe monad can be used to create a safe wrapper for functions with restricted domains (logarithm, square root...). Such functions can be composed and will still be safe without the need to perform further verifications. They are also used to simplify nested "if then" expressions that otherwise would be unwieldy and illegible. It is also implemented natively in the `Haskell` lookup tables.

The conclusion of this section should be that category theory, and specifically monads, are useful not just in mathematical applications. Studying them is not only positive for other mathematical fields, but also for the real world. It could be nourishing for an engineer to learn Haskell and, in doing so, learning category theory.

Category theory helps understand the foundations of computational languages and fuels new programming paradigms that are used in important companies such as Facebook.

4 Lawvere theories

Lawvere theories, otherwise known as algebraic theories, first appeared in F. W. Lawvere's doctoral dissertation in 1963. They are a categorical method for doing universal algebra. Universal (or equational) algebra is the study of algebraic structures focusing on their logical signature (given by operations that satisfy equational axioms). Lawvere took this idea into the categorical world: any algebraic structure can be thought of as a small category with finite products together with a functor that infuses certain properties.

Given a category $\mathcal{C}$, a *skeleton* of $\mathcal{C}$ is a full subcategory $\mathcal{C}_0 \subseteq \mathcal{C}$ which contains exactly one object of each isomorphism class of objects of $\mathcal{C}$. We denote by $\aleph_0$ a skeleton of the category of finite sets, whose objects are the finite cardinals and whose morphisms are all functions between them.

Definition 4.1. A *Lawvere theory* is a pair $L = (\mathcal{L}, I)$ where $\mathcal{L}$ is a small category with $\mathrm{Ob}(\mathcal{L}) = \mathrm{Ob}(\aleph_0)$ and with (strictly associative) finite products, and $I\colon (\aleph_0)^{\mathrm{op}} \to \mathcal{L}$ is a functor such that

- I strictly preserves finite products, and
- I is the identity in objects.

Hence every function $f\colon m \to n$ in $\aleph_0$ yields an element $If \in \mathcal{L}(n, m)$ through the (contravariant) functor I.

Lawvere theories form a category Law whose morphisms $(\mathcal{L}, I) \to (\mathcal{L}', I')$ are functors $F\colon \mathcal{L} \to \mathcal{L}'$ that are strict product-preserving functors such that $I' = FI$.

Example 4.2. The theory of groups is a Lawvere theory where $\mathcal{L}(n, m)$ is the set of group homomorphisms from a free group F_m on m generators to a free group F_n on n generators. Thus, by the universal property of free groups, $\mathcal{L}(n, m)$ is in bijective correspondence with the set of functions from m to F_n, that is, the set of m-tuples of elements in F_n.

A group is a set G equipped with an associative binary operation $\mu\colon G \times G \to G$ and a fixed element $e \in G$ acting as a unit for μ, in which every element has an inverse. Each functor $\tilde{G}\colon \mathcal{L} \to \mathsf{Set}$ preserving finite products yields a group, namely $G = \tilde{G}(1)$ equipped with the multiplication given by $\tilde{G}(\mu) \in \mathsf{Set}(G \times G, G)$ where $\mu \in \mathcal{L}(2, 1)$ sends the generator $t \in F_1 = F(t)$ to the element $xy \in F_2 = F(x, y)$. Associativity of μ comes from associativity in F_2. The unit element in G comes from $1 \in F_2$, and the existence of inverses follows from the inverses in F_2.

Example 4.3. The theory of commutative algebras over a field k is a Lawvere theory where $\mathcal{L}(m, n)$ is the set of k-algebra homomorphisms from the polynomial ring $k[x_1, \ldots, x_m]$ on m variables into the polynomial ring on n variables. The addition and multiplication of a k-algebra A come from the maps $t \mapsto x + y$ and $t \mapsto xy$ respectively from $k[t]$ to $k[x, y]$, which are elements of $\mathcal{L}(2, 1)$.

Example 4.4. Not every mathematical structure can be thought as an algebraic theory —there is no theory of fields, for example. From the point of view of universal algebra, not all properties of fields can be expressed as equational laws. Inverse elements only exist for non-zero elements, and this particularity cannot be expressed in terms of operations and equational laws. Hence there is no "free field generated by a set", which means that there does not exist any monad on Set whose algebras are precisely the fields. We will prove later that there is a relation between monads on sets and Lawvere theories, which implies that there cannot be a Lawvere theory for the theory of fields.

4.1 Models of Lawvere theories

As in the case of the theory of groups, the structures modeled by a Lawvere theory can be recovered by considering the following notion.

Definition 4.5. A *model* of a Lawvere theory $(\mathcal{L}, I)$ in a category $\mathcal{C}$ with finite products is a functor $M\colon \mathcal{L} \to \mathcal{C}$ that preserves finite products (up to isomorphism).

A model, roughly speaking, is anything that satisfies the equations of the algebraic theory, i.e., an "instance" of such.

A general fact about Lawvere theories is that for each of them, say L, there is a monad T_L such that the category of models of L is equivalent to the category of T_L-algebras. This fact will be proved below. Furthermore, as we next show, the correspondence sending L to T_L defines an equivalence of categories between the category of Lawvere theories and a full subcategory of the category of monads on Set, namely the category of *finitary* monads.

Definition 4.6. A monad T on sets is called *finitary* if it is determined by its values on the finite cardinals $n \in \aleph_0$.

More precisely, T is finitary if and only if, for every set X, the set TX is equal to the union of the subsets $(T\varphi)(Tn)$ for all functions $\varphi\colon n \to X$ and every finite ordinal n.

For a Lawvere theory $L = (\mathcal{L}, I)$, we denote by $\mathsf{Mod}\, L$ the category of models of L in Set. Since I is bijective on objects and contravariant, every $n \in \aleph_0$ yields such a model, namely

$$F_L n = \mathcal{L}(n, -);$$

that is, for each $k \in \aleph_0$ we pick $(F_L n)(k) = \mathcal{L}(n, k)$, and for each function $f\colon k_1 \to k_2$ in $\aleph_0$ we pick the function $(If)^*\colon \mathcal{L}(n, k_2) \to \mathcal{L}(n, k_1)$. This functor $\aleph_0 \to \mathsf{Mod}\, L$ extends uniquely to a functor $F_L\colon \mathsf{Set} \to \mathsf{Mod}\, L$ as follows. For each set X we define $(F_L X)(k)$ to be a quotient of the union

$$\bigcup_{n \in \aleph_0} \mathcal{L}(n, k) \times X^n$$

where, for each function $f\colon k_1 \to k_2$ in $\aleph_0$ and all $g \in \mathcal{L}(n, k_2)$ and $x \in X^{k_2}$, we identify (g, x) with $((If)^* g, \tilde{f}(x))$, where $\tilde{f}\colon X^{k_2} \to X^{k_1}$ is determined by f. For

example, if $f\colon 3 \to 5$ is the function given by $f(0) = 0$, $f(1) = 4$, $f(2) = 4$, then $\tilde{f}\colon X^5 \to X^3$ is given by $\tilde{f}(a, b, c, d, e) = (a, e, e)$.

For instance, if L is the theory of groups, then $F_L X$ can be viewed as the free group on the set X (where X need not be finite). For every word $w \in F_n = F(t_1, \ldots, t_n)$ and each $x = (x_1, \ldots, x_n) \in X^n$, we let $w(x_1, \ldots, x_n)$ be the corresponding word evaluated on the elements $x_1, \ldots, x_n$.

The functor F_L is left adjoint to the forgetful functor $U\colon \mathsf{Mod}\, L \to \mathsf{Set}$, since every function $\alpha\colon X \to M(1)$ where M is a model of L can be lifted uniquely to a natural transformation $\tilde{\alpha}\colon F_L X \to M$ by defining its nth component at $k = 1$,

$$\mathcal{L}(n, 1) \times X^n \to M(1), \tag{4.1}$$

by sending each $(\varphi, x_1, \ldots, x_n)$ to $(M\varphi)(\alpha(x_1), \ldots, \alpha(x_n))$. Hence, $T_L = UF_L$ is a finitary monad.

This is part of the proof of the following central result.

Theorem 4.7. *There exists an equivalence between the category of Lawvere theories and the category of finitary monads on sets.*

Proof. Given a Lawvere teory $L = (\mathcal{L}, I)$, the associated monad T_L is the one resulting from the adjunction $F_L\colon \mathsf{Set} \leftrightarrows \mathsf{Mod}\, L\colon U$ discussed above. As a consequence of this definition, the monad $T_L = UF_L$ preserves directed unions (i.e., it is finitary).

Conversely, given a monad T, we can associate a Lawvere theory $L_T = (\mathcal{L}, I)$ to it by defining its sets of morphism as follows:

$$\mathcal{L}_T(n, m) = \mathrm{Kl}^{\mathrm{op}}(T)(n, m) = \mathrm{Kl}(T)(m, n) = \mathsf{Set}(m, Tn).$$

Next we check that there are natural isomorphisms $L_{T_L} \cong L$ for every Lawvere theory L, and $T_{L_T} \cong T$ for every finitary monad T.

On one hand,

$$\mathcal{L}_{T_L}(n, m) = \mathsf{Set}(m, T_L n) = \mathsf{Set}(m, UF_L n) = \mathsf{Set}(m, (F_L n)(1)) =$$

$$\mathsf{Set}(m, \mathcal{L}(n, 1)) \cong \mathcal{L}(n, 1) \times \overset{m}{\cdots} \times \mathcal{L}(n, 1) \cong \mathcal{L}(n, m),$$

where the first bijection comes from the fact that $m \cong 1 + \overset{m}{\cdots} + 1$ in $\aleph_0$, and the second bijection comes from the fact that $m \cong 1 \times \overset{m}{\cdots} \times 1$ in $\aleph_0^{\mathrm{op}}$.

On the other hand, for every $n \in \aleph_0$,

$$T_{L_T} n = UF_{L_T} n = (F_{L_T} n)(1) = \mathcal{L}_T(n, 1) = \mathsf{Set}(1, Tn) \cong Tn.$$

Since both T_{L_T} and T are finitary, we may conclude that $T_{L_T} \cong T$. This completes the proof. □

We conclude this section by proving that the models of a theory can be recovered from the associated monad.

Theorem 4.8. *For every Lawvere theory $L = (\mathcal{L}, I)$ there is an equivalence between the category of models of L in* Set *and the category of T_L-algebras.*

Proof. We need to define functors $\Phi\colon \mathsf{Mod}\, L \to \mathsf{Set}^{T_L}$ and $\Psi\colon \mathsf{Set}^{T_L} \to \mathsf{Mod}\, L$ together with natural isomorphisms $\Psi\Phi \cong \mathrm{Id}_{\mathsf{Mod}\, L}$ and $\Phi\Psi \cong \mathrm{Id}_{\mathsf{Set}^{T_L}}$.

For a T_L-algebra $\mathbb{A} = (A,\, a\colon T_L A \to A)$, denote by $a_n\colon \mathcal{L}(n,1) \times A^n \to A$ the nth component of a. Pick the only model $M\colon \mathcal{L} \to \mathsf{Set}$ such that $M(1) = A$ and $M\varphi\colon A^n \to A$ is given by $M\varphi = a_n(\varphi, -)$ for each $\varphi \in \mathcal{L}(n,1)$, and define $\Psi\mathbb{A} = M$. Conversely, for a model $M\colon \mathcal{L} \to \mathsf{Set}$, define ΦM to be

$$(M(1),\, U\varepsilon_M\colon T_L M(1) \to M(1)),$$

where $\varepsilon_M\colon F_L M(1) \to M$ is the counit of the adjoint pair (F_L, U). The equality

$$U\varepsilon_M \circ \eta_{UM} = \mathrm{id}_{UM}$$

follows from the fact that ε_M is adjunct to the identity of UM. We also note that the nth component

$$(U\varepsilon_M)_n\colon \mathcal{L}(n,1) \times M(n) \to M(1)$$

sends each (φ, x) to $(M\varphi)(x)$ as a special case of (4.1).

Now, on one hand, $\Psi\Phi M$ is the only model whose value at 1 is $M(1)$ and whose value at $\varphi \in \mathcal{L}(n,1)$ is $(U\varepsilon_M)_n(\varphi, -) = M\varphi$. Hence $\Psi\Phi M = M$ for every M in $\mathsf{Mod}\, L$.

On the other hand, $\Phi\Psi\mathbb{A} = \Phi M$ where $M(1) = A$ and $M\varphi = a_n(\varphi, -)$. Thus $(U\varepsilon_M)_n = a_n$ for all n and therefore $U\varepsilon_M = a$. This implies that $\Phi M = \mathbb{A}$, as desired. □

4.2 An application

A monad $T = (T, \mu, \eta)$ on a category $\mathcal{C}$ is called *idempotent* if $\mu\colon TT \to T$ is an isomorphism. Idempotent monads are also called *localizations*. Thus, a localization on a category $\mathcal{C}$ consists of a functor $E\colon \mathcal{C} \to \mathcal{C}$ equipped with a unit $\eta\colon \mathrm{Id}_{\mathcal{C}} \to E$ such that $\eta_{EX}\colon EX \to EEX$ and $E\eta_X\colon EX \to EEX$ are equal and they are inverses of μ_X (since $\mu_X \circ \eta_{EX} = \mathrm{id}_{EX}$ and $\mu_X \circ E\eta_X = \mathrm{id}_{EX}$). General properties of localizations can be found in [2].

For a Lawvere theory $L = (\mathcal{L}, I)$, we consider models of L in the category Ab of abelian groups and their interaction with localizations on Ab. We focus on the following example: for a ring R with 1, consider the theory with

$$\mathcal{L}(m,n) = \mathsf{Ab}(R^n, R^m),$$

whose models are precisely the R-modules. A localization $E\colon \mathsf{Ab} \to \mathsf{Ab}$ is said to *lift* to R-modules if there is a localization $\tilde{E}$ on R-modules such that $U\tilde{E} \cong EU$, where U is the forgetful functor sending every R-module to the underlying abelian group —in other words, the underlying abelian group of $\tilde{E}M$ is naturally isomorphic to EUM for every R-module M.

The following is a new proof, using the language of Lawvere theories, of a result contained in [3]. The general principle behind it is the fact that a localization E lifts to T-algebras for a monad T if and only if T preserves E-equivalences (i.e., morphisms $f\colon X \to Y$ such that $Ef\colon EX \to EY$ is an isomorphism).

As we next show, the latter condition holds if T is the monad associated with the Lawvere theory of R-modules for a ring R with 1.

Theorem 4.9. *Every localization E on abelian groups lifts uniquely to R-modules for every ring R with* 1*.*

Proof. Consider the Lawvere theory $L = (\mathcal{L}, I)$ with $\mathcal{L}(m,n) = \mathsf{Ab}(R^n, R^m)$ and the associated monad T_L. Since L is enriched in abelian groups, we may view T_L as a monad on Ab.

According to [3, Theorem 4.2], it suffices to show that T_L preserves E-equivalences for every localization E. Since $T_L A$ is a natural quotient of a direct sum of $\mathcal{L}(n,1) \times A^n$ over all finite ordinals n, it suffices to check that $A^n \to B^n$ is an E-equivalence whenever $A \to B$ is an E-equivalence, and this follows from the fact that localizations commute with finite products [2, Lemma 3.5]. □

Example 4.10. As an explicit example, let $R = \mathbb{Z}_{(p)}$ be the ring of integers localized at a prime ideal (p). Then the natural quotient $A \to A/TA$ of every abelian group by its torsion subgroup defines a localization on Ab that lifts to $\mathbb{Z}_{(p)}$-modules as $A \to A/T_pA$ when A is a $\mathbb{Z}_{(p)}$-module (i.e., an abelian group uniquely q-divisible for all primes $q \neq p$). Also the rationalization functor $A \to \mathbb{Q} \otimes A$ that turns A into a uniquely divisible abelian group lifts to $\mathbb{Z}_{(p)}$-modules with the same form $A \to \mathbb{Q} \otimes A$.

5 Cointroduction

The equivalence between Lawvere theories and finitary monads is only a special case of a more general relation. First of all, arbitrary monads (not necessarily finitary) correspond to algebraic theories of a more general nature, for which the set of objects is larger than $\aleph_0$. Details about this claim can be found in [6] and [7].

Even more generally, the concept of *algebraic theory with arities* is discussed in [1], where an equivalence is given with a suitable category of *monads with arities*. In this context it is feasible to consider algebraic theories enriched in additive categories as a special case of algebraic theories with arities, so that the associated monads are defined on the category of abelian groups.

Theorem 4.9 probably holds for such algebraic theories in full generality. We expect to develop this idea in subsequent work.

One necessary ingredient to undertake this project is the concept of a colimit (and related notions of coend and left Kan extension), which we have omitted in this work for simplicity, although they are the most natural way to define finitary left adjoints of forgetful functors, such as the functor F_L discussed in Section 4 for a Lawvere theory L. The existence of left adjoints (without an explicit construction) is also guaranteed, under suitable assumptions, by general results such as the Freyd Adjoint Functor Theorem [8, p. 117].

References

[1] C. Berger, P. A. Melliès, M. Weber. Monads with arities and their associated theories. Journal of Pure and Applied Algebra **216** (2012), 2029–2048.

[2] C. Casacuberta. On structures preserved by idempotent transformations of groups and homotopy types. In: Crystallographic Groups and Their Generalizations (Kortrijk, 1999), Contemporary Mathematics 262, American Mathematical Society, Providence, RI, 2000, pp. 39–68.

[3] C. Casacuberta, O. Raventós, A. Tonks. Comparing localizations across adjunctions. `arXiv:1404.7340`

[4] M. Giry. A categorical approach to probability theory. In: Categorical Aspects of Topology and Analysis, Lecture Notes in Mathematics 915. Springer, 1982, pp. 68–85.

[5] R. Godement. Topologie algébrique et théorie des faisceaux. Hermann, Paris, 1973.

[6] M. Hyland, J. Power. The category theoretic understanding of universal algebra: Lawvere theories and monads. Electronic Notes in Theoretical Computer Science **172** (2007), 437–458.

[7] F. E. J. Linton. Some aspects of equational categories. In: Conference on Categorical Algebra (La Jolla, 1965). Springer, 1966, pp. 84–94.

[8] S. Mac Lane. Categories for the Working Mathematician, Graduate Texts in Mathematics 5, Springer, New York, 1971.

[9] E. Moggi. Notions of computation and monads. Information and Computation **93** (1991), 55–92.

[10] `https://code.facebook.com/posts/745068642270222/fighting-spam-with-haskell/`

[11] `http://www.well-typed.com/pr/2010-05-05-air_traffic_analysis_tool_for_nats/`