Free Constructions and Monadicity

by

Chentian Wu

Submitted to the Department of Mathematics in partial fulfillment of
the requirements for the degree of

Bachelor of Science with Honors

(Mathematics)

at the

University of Wisconsin–Madison

2025

Thesis supervisor:
Marco Paviotti, Assistant Professor, School of Computing, University of Kent
Uri Andrews, Professor, Department of Mathematics, University Wisconsin-Madison

Acknowledgements

I would first and foremost like to express my deepest gratitude to my advisor, Prof. Marco Paviotti. When I decided to pursue an honors thesis in Category Theory, finding a mentor was challenging due to the interdisciplinary nature of the topic; few local experts bridged the gap between formal Category Theory and PL semantics.

Despite the distance between the University of Kent and Madison, and my relative inexperience in the field, Prof. Paviotti graciously agreed to supervise me. I am immensely grateful for his willingness to take a chance on me. Our regular meetings since January 2025 have been a constant source of direction, providing the intuition and resources necessary for this work. Without his mentorship, I would not have discovered the profound joy of applying categorical methods to programming languages.

My sincere thanks also go to Prof. Uri Andrews at UW-Madison for serving as my second advisor. I am also indebted to my Programming Language instructors at UW-Madison, whose high-quality teaching provided the essential foundation for this thesis.

Finally, I thank my parents for their unwavering support, most significantly in my decision to transfer to UW-Madison to pursue my academic goals.

Chentian Wu
Madison, Wisconsin
Dec 25, 2025

Contents

Abstract

This thesis investigates the relationship between free constructions and monadicity, uniting mathematics and theoretical computer science through the lens of Category Theory.

A free construction, intuitively, is the *cheapest* way of endowing an object with structure. Here, *cheapest* means that the object possesses exactly the necessary properties – no more and no less – to support the desired structure. This can be understood through the principle of "no junk and no noise". The expression "no junk" means that the object contains nothing beyond what arises from a chosen set of generators from the original object, while "no noise" means that no additional equations are imposed beyond those required by the structure itself.

Free constructions appear across diverse contexts, from free groups in algebra to free monads that model computational effects in programming languages theory. Category theory provides a unified framework in which both algebraic structures and computational phenomena arise from the so-called free-forgetful adjunctions, which in turn generate monads capturing the essence of these theories. The central question explored here is: When is the category of models of a theory equivalent to the category of algebras for its associated monad? The answer is provided by the notion of monadicity, formalized in Barr-Beck's Monadicity Theorem. Furthermore, this equivalence turns out to have profound implications in universal algebra and even in theoretical computer science.

The thesis is structured as follows:

1. Chapter 1 offers an intuitive introduction to the key ideas.

2. Chapters 2 to 4 develop the categorical foundations in detail, covering adjunctions, monads, and the monadicity theorems.

3. Chapter 5 applies this framework to algebraic theories, demonstrating that their categories of models are indeed monadic.

Chapter 1

Introduction

The history of mathematics is rarely a linear procession of orderly discoveries. Instead, it is often a tumultuous convergence of disparate streams—topology, algebra, geometry, and logic—crashing together to form new, unified landscapes. The subject of this thesis, *Free Constructions* and *Monadicity*, is one of the profound examples of this phenomenon in the 20th century.

To the contemporary student of Category Theory, Beck's Monadicity Theorem is often presented as a pristine technical criterion: a set of necessary and sufficient conditions for a functor to be monadic. Similarly, to the computer scientist, the Monad is often viewed merely as a design pattern for managing side effects. However, these modern crystallizations strip away the decades of intellectual struggle that birthed them.

This thesis investigates the deep structural link between the process of generating structure, known as *freeness*, and the process of recognizing structure, referred to as *monadicity*. To understand why these concepts are central to both mathematics and computer science, we must first understand the historical crises that necessitated them.

Historical Origins

The story begins in the 1950s, a period dominated by the rapid ascendancy of homological algebra. The central problem of this era was the computation of cohomology groups, which measures the topological or algebraic *holes* in a topological space or algebraic structure. While the tools developed by Henri Cartan and Samuel Eilenberg [CE99] were powerful, they faced a significant limitation known as the *additive bias*—the machinery was strictly confined to linear and abelian categories.

The Standard Construction. In the years following World War II, the French school of mathematics revolutionized topology through the invention of *sheaf theory*. A sheaf tracks local data across a global space. However, computing the cohomology of a sheaf required an *injective resolution*—a sequence of embeddings into injective objects. In the category of sheaves, injective objects are monstrously complex, *large,* and opaque. The theory was elegant, but the machinery required to compute it was intractable.

The breakthrough came in 1958 with Roger Godement's treatise on algebraic topology. Godement [God58] introduced a radical new method for constructing resolutions based on *flasque* (flabby) sheaves. Unlike injectives, flabby sheaves had a concrete geometric definition. More importantly, Godement discovered a canonical way to embed *any* sheaf into a flabby sheaf via a functor $\mathbf{T}$. This construction came equipped with natural maps that allowed one to iterate the process, generating a sequence:

$$\mathcal{F} \to \mathbf{T}\mathcal{F} \to \mathbf{T}\mathbf{T}\mathcal{F} \to \ldots$$

This sequence, which Godement termed the *standard construction*, was guaranteed to be acyclic, allowing for the computation of cohomology without ever appealing to opaque injective objects.

Structurally, Godement's construction consisted of an endofunctor $\mathbf{T}$ equipped with two natural transformations: a *unit* η that embedded the sheaf into the construction, and a *multiplication* μ that collapsed the iteration. The community began to realize that the *standard construction* was not a trick for sheaves; it was a universal mechanism for resolving structure.

The Formalization of Triples. The transition from these ad-hoc constructions to a formal theory was catalyzed by the discovery of *adjoint functors* by Daniel Kan in 1958 [Kan58]. It took several years for the community to realize the profound connection: every *standard construction* was the shadow cast by an adjunction.

This realization was formalized in the seminal 1965 paper by Samuel Eilenberg and John C. Moore, *Adjoint functors and triples* [EM65]. They demonstrated that every adjoint pair $\mathbf{F} \dashv \mathbf{U}$ gives rise to a *triple*,

$$\mathbb{T} = (T, \eta, \mu)$$

which is now called a *monad*, on the domain category. The unit of the adjunction η becomes the *unit* of the monad, and the counit yields the multiplication μ.

This simple observation changed the landscape of algebra. It meant that every time a mathematician defined a *free* object—a free group, a free ring, or a polynomial algebra—they were implicitly creating a monad. Eilenberg and Moore went further, asking the inverse question: Given a monad $\mathbb{T}$, can we reconstruct the adjunction that generated it? Their answer was the construction of the *Eilenberg-Moore Category* $\mathcal{A}^{\mathbb{T}}$, the category of *algebras* over the monad. This unified the concept of *algebraic structure* under a single banner.

Beck's Insight. This brings us to the central figure of the theoretical development: Jonathan Beck. Working in the mid-1960s, Beck sought to shatter the additive bias of homological algebra. He wanted to define a cohomology theory that applied universally to *any* algebraic structure—groups, Lie algebras, and abstract monoids. To legitimize his new *Triple Cohomology*, Beck needed to prove it matched classical results. Specifically, in the classical theory of rings, the second cohomology group classifies *singular extensions*. Beck faced a critical challenge: he needed to treat the messy, concrete category of *singular extensions over X* as a category of algebras itself. He needed to show that the operations of forming extensions behaved exactly like the operations of an algebra over a monad.

This necessity birthed the *Monadicity Theorem*, which was originally known as the *Tripleability Theorem*. Beck identified that algebraic structure is essentially about *coequalizers*. In algebra, we define structures using generators and relations. Categorically, generators correspond to free objects, and relations correspond to coequalizers that capture quotients. Beck's insight, that the coequalizer is the categorical essence of an equation, provided the rigorous authorization to treat nonlinear categories using the powerful machinery of the Eilenberg-Moore category. It is worth noting that these same conditions would later prove fundamental in algebraic geometry, specifically in characterizing the effectiveness of *descent* for gluing local geometric structures [BR70].

Alternative Perspectives: Lawvere and Linton. Parallel to the development of monadicity, F.W. Lawvere introduced a distinct categorical framework for universal algebra in 1963 [Law63]. Rather than focusing on endofunctors, Lawvere defined an *algebraic theory* as a small category $\mathcal{L}$ with finite products, where the objects are natural numbers representing arities. In this framework, an algebra is simply a product-preserving functor from $\mathcal{L}$ to the category of sets. This approach offered a coordinate-free, syntax-independent definition of algebraic structure that elegantly handled finitary operations. In 1966, Linton demonstrated that the category of algebras for a finitary monad is equivalent to the category of models for a Lawvere theory [Lin66]. Linton's work clarified that while Lawvere theories were conceptually cleaner for classical finitary algebra, the monadic approach was more general, capable of handling infinitary structures, such as compact Hausdorff spaces, where operations have infinite arity. This synthesis ensured that algebraic structure was a unified concept, regardless of the categorical tool used to describe it.

Moggi and Computational Effects. The bridge to computer science was built by Eugenio Moggi in 1991. Moggi observed that *impure* features—like modifying memory state, raising exceptions, or non-determinism—could be modeled by wrapping the return type of a function in a monad $\mathbb{T}$ [Mog88, Mog91]. For example, the *State Monad* models computations that read and write to a store. The *List Monad* models non-determinism. This insight revolutionized functional programming by showing that the abstract machinery of monads could encapsulate effectful behavior within a pure functional language.

Algebraic Effects. While Moggi established that computational effects can be modeled by monads, this approach faces a significant practical limitation: monads do not compose naturally. The combination of two monads is not necessarily a monad, making it difficult to combine different effects in a modular way. Gordon Plotkin and John Power addressed this by grounding effects in the theory of *Algebraic Effects* [PP02, PP03]. In this view, an effect is not an opaque functor but is defined by *primitive commands* and *laws* that govern their behavior. Crucially, algebraic theories compose easily via the sum of their signatures and the union of their equations. This perspective returns us to the free construction: operations generate a *free monad* that represents raw syntax, and equations impose a quotient that captures the logic of the effect.

Free Monad: An Intuitive Introduction

Free constructions often appear with a *universal* property, which is intuitively the best or most general way to impose some structure. In this section, we develop an intuitive understanding of *free monad*, which is one kind of free construction for obvious reasons. Free monads are used in functional programming languages like Haskell to represent syntax of a computational effect.

Signatures as functors. To understand how computational effects can be treated as algebraic structures, we must first formalize the notions of operations and laws using the language of category theory. The starting point for any algebraic theory is a *signature*, which specifies the interface of the algebraic structure.

A signature Σ is a collection of pairs $(\mathsf{op}_i, \mathsf{ar}_i)_{i\in I}$, where each op_i is called an operation symbol and $\mathsf{ar}_i \in \mathbb{N}$ is called its arity. An endofunctor $\mathbf{\Sigma}$ on $\mathbf{Set}$ generalizes the concept of a signature, that is, for a signature having a set of operation symbols a_i of arity i (for each $i \geq 0$), we define a functor $\mathbf{\Sigma} : \mathbf{Set} \to \mathbf{Set}$ by the polynomial formula:

$$\mathbf{\Sigma} X = a_0 + a_1 \times X + a_2 \times X^2 + \cdots + a_n \times X^n \tag{1.1}$$

where a_i is a finite set indexing the i-ary operations.

This endofunctor $\mathbf{\Sigma}$ captures the shape of *one-level* Σ-operations: given a set X of inputs, $\mathbf{\Sigma} X$ is the set of all single-step computations one can build—either a nullary operation (there are a_0 ways to do that) or an i-ary operation with choices of i inputs from X.

Example 1.1 (The Signature of Monoids)**.** Consider the theory of monoids, Mon. The signature is given by $\Sigma_{\mathsf{Mon}} = \{(*, 2), (e, 0)\}$, consisting of a binary multiplication and a nullary unit element. Categorically, this corresponds to the functor:

$$\mathbf{\Sigma}_{\mathsf{Mon}}(X) = 1 + X^2$$

Here $a_0 = 1$ represents the constant e, and $a_2 = 1$ represents the binary operation $*$.

Syntax as free monads. From a signature Σ, we construct the *raw syntax* or terms. Let $\Gamma = \{x_1, \ldots, x_n\}$ be a context of variables. The set $\Sigma^*(\Gamma)$ of all well-formed Σ-terms is defined inductively: every variable is a term, and if op is an n-ary operation, then $\mathsf{op}(t_1, \ldots, t_n)$ is a term. That is, terms are trees where leaves are variables and internal nodes are operations. From the endofunctor perspective, the set of terms $\Sigma^*(\Gamma)$ is a solution of the set equation up to isomorphism:

$$\Sigma^*(\Gamma) \cong \Gamma + \mathbf{\Sigma}\Sigma^*(\Gamma) \tag{1.2}$$

and one solution is given by the *least fixed point*:

$$\Sigma^*(X) = \mu Y. X + \mathbf{\Sigma} Y$$

This construction lifts $\mathbf{\Sigma}$ into another functor $\mathbf{\Sigma}^* : \mathbf{Set} \to \mathbf{Set}$, where $\mathbf{\Sigma}^* X = \Sigma(X)$ is the set of Σ-terms with variables drawn from X. There are two natural structures that come with this functor:

1. The *injection* $\eta_X : X \to \mathbf{\Sigma}^* X$, which embeds variables as trivial terms.
2. The *flattening* $\mu_X : \mathbf{\Sigma}^*\mathbf{\Sigma}^* X \to \mathbf{\Sigma}^* X$, which performs substitution. It takes a *term-of-terms* and flattens it into a single term by replacing variables with the terms they hold.

And in fact, for readers familiar with Haskell, this construction is exactly the `Free` monad over the functor $\mathbf{\Sigma}$. Equation (1.2) could be implemented using an algebraic data type in Haskell,

```haskell
data Free (f :: Type -> Type) a
    = Pure a
    | Op (f (Free f a))
```

which directly corresponds to the fixed point isomorphism above: a term is either a pure variable from Γ or an operation applied to sub-terms from $\Sigma\Sigma^*(\Gamma)$. Here, `Pure` corresponds to the injection of variables η_X, and `Op` wraps one layer of the functor `f`, inductively building the tree structure of terms.

We can show that this is indeed a monad by defining the monadic operations. we must define the binding operation, which corresponds to syntactic substitution.

```haskell
instance (Functor f) => Monad (Free f) where
    return = Pure
    (Pure x) >>= g = g x
    (Op y) >>= g = Op $ fmap (>>= g) y
```

The `return` function simply lifts a value into a leaf node. The bind operator `(>>=)` takes a syntax tree and a function `g` that maps variables to new syntax trees. It traverses the structure: if it encounters a variable `Pure x`, it replaces it with the tree generated by `g x`. If it encounters an operation `Op`, it recursively applies the substitution to the children. This is precisely the implementation of the monad multiplication μ, which grafts new trees onto the leaves of an existing tree.

Notably, a `Free` monad is indeed *free*, in the sense that there is a universal property of $\mathbf{\Sigma}^* X$. This computational view aligns perfectly with the categorical definition. The free monad $\mathbf{\Sigma}^*$ satisfies a vital *universal property*: it is the *most general* monad generated by the functor $\mathbf{\Sigma}$. Formally, for any monad $(\mathbf{T}, \eta^T, \mu^T)$ and any natural transformation $\alpha : \mathbf{\Sigma} \to \mathbf{T}$, there exists a unique monad morphism $\hat{\alpha} : \mathbf{\Sigma}^* \to \mathbf{T}$ such that

$$\hat{\alpha} \circ \eta = \eta^{\mathbf{T}}$$

for all sets X.

Thesis Structure

This thesis explores the mathematical machinery that enables this unification. We will journey from the basic definitions of categories to the advanced criteria of the Monadicity Theorem, culminating in their application to algebraic theories.

Chapter 2. The Basic Language establishes the foundational vocabulary of Category Theory. We introduce functors, natural transformations, and the crucial concept of *Adjunctions*, which generate free constructions. We also detail the theory of limits and colimits, specifically *Coequalizers*, which act as the categorical generalization of equations.

Chapter 3. Monads and Algebras formally defines the Monad and its associated algebras. We explore the two canonical resolutions of a monad: the *Kleisli Category* (representing free algebras/syntax) and the *Eilenberg-Moore Category* (representing all algebras/semantics). We show how the Free Construction naturally leads to these structures.

Chapter 4. Monadicity Theorems forms the technical core of the thesis. We present *Beck's Monadicity Theorem* and its variants. We analyze the subtle distinctions between "Weak," "Crude," and "Precise" monadicity. A significant portion of this chapter is dedicated to the role of *Coequalizers*—specifically split, reflexive, and absolute coequalizers—in determining when a functor is monadic. We demonstrate that the "exactness" conditions of the theorem are precisely what is needed to reconstruct the algebraic structure from the monad.

Chapter 5. Algebraic Theories applies this framework to the study of algebraic structures. We prove the correspondence between equational theories and Monads. We show that the category of models for an algebraic theory is strictly monadic over the category of sets. Finally, we connect this back to *Algebraic Effects*, illustrating how computational notions arise as algebras for a theory defined by operations and equations.

In summary, this thesis demonstrates that the seemingly abstract machinery of Monadicity is, in fact, the rigorous bridge between Syntax and Semantics. Whether in the context of mathematical homology or programming language effects, monadicity realizes the insight that structure is not accidental—it is generated by operations, governed by equations, and characterized by the preservation of exactness.

Chapter 2

The Basic Language

Category theory provides a unifying language for formalizing relationships between mathematical structures, often with a hierarchical character. Usually one begins with the basic notion of a category, and relations between categories are expressed by functors. Furthermore, we can study relations between these functors, called natural transformations. These ideas can be extended to higher-dimensional structures, but in this thesis we work only with categories, functors, and natural transformations; Section 2.1 introduces these notions and their basic properties.

Freeness captures a certain *best* or *universal* aspect of constructions, and category theorists formalize this through universal properties. We will focus on universal properties in Section 2.2, especially those whose internal universal character is closely tied to freeness, most notably adjunctions (Section 2.3).

2.1 Categories, Functors and Natural Transformations

A *category* is the basic building block of Category Theory. It is designed to be abstract enough, with only a few essential constraints, to capture the general essence of mathematical structures and the relationships between them. From the perspective that mathematicians prefer structures that are beautiful and well-behaved, categories distill the most essential properties of such structures. In this section, we study the basic notions and properties of categories and functors, which describe transitions or mappings between categories. Finally, we introduce natural transformations, which are higher-level transitions between functors.

Definition 2.1 (Category)**.** A *category* $\mathcal{A}$ consists of

(1) A collection of *objects*, denoted $\operatorname{Ob}\mathcal{A}$

(2) For each pair of objects $A, B \in \operatorname{Ob}\mathcal{A}$, a collection of *morphisms* (or arrows) from A to B, denoted $\operatorname{Hom}_{\mathcal{A}}(A, B)$ or $\mathcal{A}(A, B)$. The collection of all morphisms in $\mathcal{A}$ is denoted $\operatorname{Hom}\mathcal{A}$.

(3) For each object $A \in \operatorname{Ob}\mathcal{A}$, an *identity morphism* $\mathrm{id}_A : A \to A$

(4) For each triple of objects $A, B, C \in \operatorname{Ob}\mathcal{A}$, a *composition law*

$$\circ : \mathcal{A}(B, C) \times \mathcal{A}(A, B) \to \mathcal{A}(A, C)$$

such that the following axioms hold:

Associativity. For morphisms $f : A \to B$, $g : B \to C$, and $h : C \to D$,

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

Identity. For any morphism $f : A \to B$,

$$f \circ \mathrm{id}_A = f = \mathrm{id}_B \circ f$$

A category $\mathcal{A}$ is called *small* if the $\operatorname{Ob}\mathcal{A}$ is a set, and for each pair of objects $A, B \in \operatorname{Ob}\mathcal{A}$, the set $\mathcal{A}(A, B)$ is also a set. A category $\mathcal{A}$ is called *large* if it is not small.

A lot of mathematical structures can be organized into categories, for example:

Set. The category of sets, where objects are sets and morphisms are functions between sets;

Grp. The category of groups, where objects are groups and morphisms are group homomorphisms;

Top. The category of topological spaces, where objects are topological spaces and morphisms are continuous functions between topological spaces.

Additional mathematical structures are described in Appendix A, with proofs of well-definedness.

Morphisms generalize the notion of homomorphisms, which are maps preserving structures of mathematical objects. In Category Theory, we also care about special morphisms that capture important properties of objects and their relationships, in particular, monomorphisms and epimorphisms and isomorphisms.

Definition 2.2 (Monomorphism, Epimorphism and Isomorphism)**.** A morphism $f : X \to Y$ in a category $\mathcal{A}$ is called a

monomorphism if for any two morphisms $g, h : Z \to X$ in $\mathcal{A}$,

$$f \circ g = f \circ h \implies g = h.$$

epimorphism if for any two morphisms $g, h : Y \to Z$ in $\mathcal{A}$,

$$g \circ f = h \circ f \implies g = h.$$

isomorphism if there exists a morphism $g : Y \to X$ such that:

$$g \circ f = \mathrm{id}_X \quad \text{and} \quad f \circ g = \mathrm{id}_Y$$

we write $X \cong Y$ and call g the *inverse* of f.

A monomorphism generalizes the notion of injective functions, while an epimorphism generalizes the notion of surjective functions. It's clear from the definition that a morphism is an isomorphism if and only if it is both a monomorphism and an epimorphism.

The *functor* is the first category-theoretic concept that is subtle and different from what we usually see in traditional mathematical structures.

Definition 2.3 (Functor)**.** A *functor* $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ from a category $\mathcal{A}$ to a category $\mathcal{B}$ consists of

(1) A mapping $\mathbf{F} : \mathrm{Ob}\,\mathcal{A} \to \mathrm{Ob}\,\mathcal{B}$

(2) A mapping $\mathbf{F} : \mathrm{Hom}\,\mathcal{A} \to \mathrm{Hom}\,\mathcal{B}$. In particular, for each pair of objects $X, Y \in \mathrm{Ob}\,\mathcal{A}$, a mapping
$$\mathbf{F} : \mathcal{A}(X, Y) \to \mathcal{B}(\mathbf{F}X, \mathbf{F}Y)$$

satisfying:

(1) Identity preservation: $\mathbf{F}\mathrm{id}_X = \mathrm{id}_{\mathbf{F}X}$ for all $X \in \mathrm{Ob}\,\mathcal{A}$

(2) Composition preservation: $\mathbf{F}(g \circ f) = \mathbf{F}g \circ \mathbf{F}f$ for all composable morphisms f, g in $\mathcal{A}$

When we speak of functors without qualification, we mean *covariant functors*, which preserve the direction of morphisms as defined above. A *contravariant functor* $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ reverses the direction of morphisms, mapping $\mathcal{A}(A, B)$ to $\mathcal{B}(\mathbf{F}B, \mathbf{F}A)$ and satisfying $\mathbf{F}(g \circ f) = \mathbf{F}f \circ \mathbf{F}g$ for composable morphisms f, g.

Example 2.1 (Forgetful Functor on $\mathbf{Grp}$)**.** The *forgetful functor* $\mathbf{U} : \mathbf{Grp} \to \mathbf{Set}$ sends each group $(G, \cdot)$ to its underlying set G and each group homomorphism $f : G \to H$ to the underlying function $f : G \to H$. This functor *forgets* the group structure, retaining only the set structure. We will encounter more general forgetful functors in later chapters.

To see that $\mathbf{U}$ is a well-defined functor, we need to verify the two functoriality properties:

(1) Identity preservation: For any group $(G, \cdot)$, the identity morphism in $\mathbf{Grp}$ is the identity group homomorphism $\mathrm{id}_G : G \to G$. Applying the forgetful functor, we have $\mathbf{U}(\mathrm{id}_G) = \mathrm{id}_G$, which is indeed the identity function on the underlying set G.

(2) Composition preservation: For any two composable group homomorphisms $f : G \to H$ and $g : H \to K$, their composition in $\mathbf{Grp}$ is given by $(g \circ f)(x) = g(f(x))$ for all $x \in G$. Applying the forgetful functor, we have $\mathbf{U}(g \circ f) = g \circ f$, which is exactly the composition of the underlying functions $\mathbf{U}f : G \to H$ and $\mathbf{U}g : H \to K$ in $\mathbf{Set}$.

Another important example of functors is the free functor, which constructs free objects in a category from objects in another category. For example, the free group functor constructs the *best* group out of a set.

Example 2.2 (Free Group Functor)**.** The *free group functor* $\mathbf{F} : \mathbf{Set} \to \mathbf{Grp}$ sends each set S to the free group $\mathbf{F}S$ generated by S, and each function $f : X \to Y$ to the unique group homomorphism $\mathbf{F}f : \mathbf{F}X \to \mathbf{F}Y$ extending f. In particular, for a given set S, define $S^{-1} = \{s^{-1} \mid s \in S\}$ as the set of formal inverses of elements in S. The free group $\mathbf{F}S$ consists of all finite words $(S \cup S^{-1})^*$ formed by elements of $S \cup S^{-1}$, subject to the equivalence relation $\sim$ induced by the group axioms

$$\begin{aligned} ss^{-1} &\sim \varepsilon, \\ s_1(s_2s_3) &\sim (s_1s_2)s_3, \\ \varepsilon s \sim s &\sim s\varepsilon. \end{aligned}$$

In $\mathbf{F}S$, ε denotes the empty word, which serves as the identity element of the group. The well-definedness of the free group requires proving confluence and termination of the reduction system, which is a standard result in combinatorial group theory and is therefore omitted here.

To see that $\mathbf{F}$ is a well-defined functor, however, is more complicated than the forgetful functor in example 2.1. We could unfold all technical details and verify the functor laws directly, but an easier way is to use the *universal property* of free groups, which states the following. For any set S and any group G, given a function $f : S \to \mathbf{U}G$, there exists a unique group homomorphism $\overline{f} : \mathbf{F}S \to G$ such that the following diagram commutes:

$$\begin{tikzcd}
S \arrow[r, "\iota_S"] \arrow[dr, "f"'] & \mathbf{F}S \arrow[d, dashed, "\overline{f}"] \\
 & G
\end{tikzcd}$$

where $\iota_S : S \to \mathbf{F}S$ is the canonical injection mapping each element $s \in S$ to the corresponding generator in $\mathbf{F}S$. Using this universal property, we can verify the functoriality of $\mathbf{F}$:

(1) Identity preservation. For any set S, the identity function $\mathrm{id}_S : S \to S$ induces a unique group homomorphism $\mathbf{F}\mathrm{id}_S : \mathbf{F}S \to \mathbf{F}S$ such that $\mathbf{F}\mathrm{id}_S \circ \iota_S = \iota_S$. By uniqueness in the universal property, we have $\mathbf{F}\mathrm{id}_S = \mathrm{id}_{\mathbf{F}S}$.

(2) Composition preservation. For any two composable functions $f : X \to Y$ and $g : Y \to Z$, the composition $g \circ f : X \to Z$ induces a unique group homomorphism $\mathbf{F}(g \circ f) : \mathbf{F}X \to \mathbf{F}Z$ such that $\mathbf{F}(g \circ f) \circ \iota_X = \iota_Z \circ (g \circ f)$. On the other hand, f and g induce unique homomorphisms $\mathbf{F}f : \mathbf{F}X \to \mathbf{F}Y$ and $\mathbf{F}g : \mathbf{F}Y \to \mathbf{F}Z$ satisfying $\mathbf{F}f \circ \iota_X = \iota_Y \circ f$ and $\mathbf{F}g \circ \iota_Y = \iota_Z \circ g$. Composing these,

$$(\mathbf{F}g \circ \mathbf{F}f) \circ \iota_X = \mathbf{F}g \circ (\mathbf{F}f \circ \iota_X) = \mathbf{F}g \circ (\iota_Y \circ f) = (\mathbf{F}g \circ \iota_Y) \circ f = (\iota_Z \circ g) \circ f = \iota_Z \circ (g \circ f).$$

By the uniqueness part of the universal property, we conclude that $\mathbf{F}(g \circ f) = \mathbf{F}g \circ \mathbf{F}f$.

There is a structure behind the universal property, called *adjunction*, which we will study in section 2.3.

Finally, natural transformations provide a way to describe the transitions between functors, concretizing the vague idea of *naturality* in mathematics.

Definition 2.4 (Natural Transformation)**.** Let $\mathbf{F}, \mathbf{G} : \mathcal{A} \to \mathcal{B}$ be functors. A *natural transformation* $\eta : \mathbf{F} \Rightarrow \mathbf{G}$ consists of a morphism $\eta_A : \mathbf{F}A \to \mathbf{G}A$ in $\mathcal{B}$ for each object $A \in \operatorname{Ob} \mathcal{A}$, such that for every morphism $f : A \to B$ in $\mathcal{A}$, $\mathbf{G}f \circ \eta_A = \eta_B \circ \mathbf{F}f$, i.e. diagram (2.1) commutes:

$$\begin{tikzcd}
\mathbf{F}A \arrow[r, "\eta_A"] \arrow[d, "\mathbf{F}f"'] & \mathbf{G}A \arrow[d, "\mathbf{G}f"] \\
\mathbf{F}B \arrow[r, "\eta_B"'] & \mathbf{G}B
\end{tikzcd} \tag{2.1}$$

If every component $\eta_A : \mathbf{F}A \to \mathbf{G}A$ is an isomorphism in $\mathcal{B}$, we call η a *natural isomorphism* and write $\mathbf{F} \cong \mathbf{G}$.

Example 2.3 (Identity Natural Transformation)**.** For any functor $\mathbf{F} : \mathcal{A} \to \mathcal{B}$, the *identity natural transformation* $\mathrm{id}_{\mathbf{F}} : \mathbf{F} \Rightarrow \mathbf{F}$ is defined by $(\mathrm{id}_{\mathbf{F}})_A = \mathrm{id}_{\mathbf{F}A}$ for all objects A.

This hierarchical structure of functors and natural transformations allows us to relax the strict notion of isomorphism between categories to a more flexible and weaker concept called equivalence of categories, which are more common in practice.

Definition 2.5 (Equivalence of Categories)**.** Two categories $\mathcal{A}$ and $\mathcal{B}$ are *equivalent*, written as $\mathcal{A} \simeq \mathcal{B}$, if there exist functors $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ and $\mathbf{G} : \mathcal{B} \to \mathcal{A}$ such that:

$$\mathbf{GF} \cong \mathbf{id}_{\mathcal{A}} \quad \text{and} \quad \mathbf{FG} \cong \mathbf{id}_{\mathcal{B}}$$

where $\mathbf{id}_{\mathcal{A}}$ and $\mathbf{id}_{\mathcal{B}}$ are the identity functors.

2.2 Universal Constructions: Limits and Colimits

This section introduces limits and colimits, which are fundamental concepts in category theory and generalize many familiar constructions from algebra and topology, including products, coproducts, equalizers, and coequalizers. For example, the product of two topological spaces (with the product topology) is a limit in **Top**. There are two noteworthy internal features of category-theoretic definitions that will appear in this section: duality and universal properties. The Duality Principle allows us to obtain the definition of colimits directly from that of limits by reversing the direction of all morphisms. Universal properties capture the essence of constructions by characterizing them through their relationships with other objects, rather than through their internal structure.

2.2.1 Diagrams and Cones

To define limits and colimits precisely, we need a few fundamental concepts: *diagrams* and *cones/cocones*. Intuitively, a diagram is a configuration of objects and morphisms in a category that we wish to study.

Definition 2.6 (Diagram)**.** Let $\mathcal{J}$ be a small category. A *diagram* of shape $\mathcal{J}$ in a category $\mathcal{A}$ is a functor $\mathbf{D} : \mathcal{J} \to \mathcal{A}$.

A cone represents a way of *looking at* this diagram from a particular vantage point: it consists of a single object together with morphisms to each object in the diagram, such that these morphisms respect the structure of the diagram, while a cocone represents a way of *assembling* the objects in the diagram into a single object, again respecting the structure of the diagram.

Definition 2.7 (Cone)**.** Let $\mathbf{D} : \mathcal{J} \to \mathcal{A}$ be a diagram. A *cone* over $\mathbf{D}$ consists of

- An object $C \in \operatorname{Ob} \mathcal{A}$ (the *apex* of the cone)
- For each object $j \in \operatorname{Ob} \mathcal{J}$, a morphism $\pi_j : C \to \mathbf{D}j$

Such that for every morphism $f : j \to k$ in $\mathcal{J}$, diagram (2.2) commutes:

$$\begin{tikzcd}
C \arrow[r, "\pi_k"] \arrow[dr, "\pi_j"'] & \mathbf{D}k \\
& \mathbf{D}j \arrow[u, "\mathbf{D}f"']
\end{tikzcd} \tag{2.2}$$

That is, $\mathbf{D}f \circ \pi_j = \pi_k$.

Definition 2.8 (Cocone)**.** Let $\mathbf{D} : \mathcal{J} \to \mathcal{A}$ be a diagram. A *cocone* under $\mathbf{D}$ consists of

- An object $C \in \operatorname{Ob} \mathcal{A}$ (the *apex* of the cocone)
- For each object $j \in \operatorname{Ob} \mathcal{J}$, a morphism $\iota_j : \mathbf{D}j \to C$

Such that for every morphism $f : j \to k$ in $\mathcal{J}$, diagram (2.3) commutes:

$$\begin{tikzcd}
\mathbf{D}j \arrow[r, "\mathbf{D}f"] \arrow[dr, "\iota_j"'] & \mathbf{D}k \arrow[d, "\iota_k"] \\
& C
\end{tikzcd} \tag{2.3}$$

That is, $\iota_k \circ \mathbf{D}f = \iota_j$.

The universal nature of limits arises when we ask: What is the *best* or most general way to look at a given diagram? Similarly, colimits arise when we ask for the *most efficient* way to assemble the objects in a diagram into a single unified structure.

2.2.2 Limits

Definition 2.9 (Limit)**.** Let $\mathbf{D} : \mathcal{J} \to \mathcal{A}$ be a diagram. A *limit* of $\mathbf{D}$, denoted by $\lim \mathbf{D}$, is a cone $(L, \{\pi_j\}_{j\in\mathcal{J}})$ over $\mathbf{D}$ that is universal in the sense that for any other cone $(C, \{\phi_j\}_{j\in\mathcal{J}})$ over $\mathbf{D}$, there exists a unique morphism $u : C \to L$ such that $\pi_j \circ u = \phi_j$ for all $j \in \mathcal{J}$.

A limit represents a class of structures, and two of them are of special interest to us: *products* and *equalizers*.

Definition 2.10 (Product)**.** Let $\mathcal{J}$ be a discrete category with two objects. A diagram $\mathbf{D} : \mathcal{J} \to \mathcal{A}$ consists of two objects A, B. A limit of this diagram is a *product* $A \times B$ together with projection maps $\pi_1 : A \times B \to A$ and $\pi_2 : A \times B \to B$.

The universal property states that for any object C with morphisms $f : C \to A$ and $g : C \to B$, there exists a unique morphism $\langle f, g\rangle : C \to A \times B$ such that $\pi_1 \circ \langle f, g\rangle = f$ and $\pi_2 \circ \langle f, g\rangle = g$. In particular, diagram (2.4) commutes:

$$\begin{tikzcd}
& C \arrow[ld, "f"'] \arrow[d, dashed, "\langle f,g\rangle"] \arrow[rd, "g"] & \\
A & A \times B \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & B
\end{tikzcd} \tag{2.4}$$

Many familiar structures are products in various categories, for example

Set. The *product* object of two sets is their cartesian product equipped with the standard projection functions.

Grp. The *product* object of two groups is their direct product equipped with the componentwise group operation and projection homomorphisms.

Top. The *product* object of two topological spaces is their cartesian product equipped with the product topology and projection continuous maps. This also gives a much clearer definition of product topology, as the coarsest topology making the projection maps continuous.

Detailed verifications that these constructions satisfy the universal property of products can be found in Appendix A.

Definition 2.11 (Equalizer)**.** Let $\mathcal{J}$ be the category with two objects and a parallel pair of morphisms between them. A diagram $\mathbf{D} : \mathcal{J} \to \mathcal{A}$ consists of two objects A, B and two morphisms $f, g : A \to B$. A limit of this diagram is an *equalizer* $\mathrm{eq}(f, g)$ together with a morphism $e : \mathrm{Eq}(f, g) \to A$ such that $f \circ e = g \circ e$.

The universal property states that for any object C with a morphism $h : C \to A$ such that $f \circ h = g \circ h$, there exists a unique morphism $u : C \to \mathrm{Eq}(f, g)$ such that $e \circ u = h$. In particular, diagram (2.5) commutes:

$$\begin{tikzcd}
\mathrm{eq}(f,g) \arrow[r, "e"] & A \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & B \\
C \arrow[u, dashed, "u"] \arrow[ur, "h"'] & &
\end{tikzcd} \tag{2.5}$$

Example 2.4 (Equalizer in R-**Mod**)**.** Given a ring R, the kernel of a ring homomorphism $f : M \to N$ between R-modules is the equalizer of f and the zero morphism $0 : M \to N$.

Proof. By R-module homomorphism theorem $\ker f \trianglelefteq M$ and the inclusion map $i : \ker f \to M$ satisfies $f \circ i = 0 \circ i$. For any R-module C with a morphism $h : C \to M$ such that $f \circ h = 0 \circ h$, the image of h lies in $\ker f$, so there exists a unique morphism $u : C \to \ker f$ such that $i \circ u = h$. □

2.2.3 Colimits

A good thing about Category Theory is duality: proving a result establishes its dual for free! In this case, we can obtain the definition of colimits directly from that of limits by reversing the direction of all morphisms.

Definition 2.12 (Colimit)**.** Let $\mathbf{D} : \mathcal{J} \to \mathcal{A}$ be a diagram. A *colimit* of $\mathbf{D}$, denoted by $\operatorname{colim} \mathbf{D}$, is a cocone $(L, \{\iota_j\}_{j \in \mathcal{J}})$ under $\mathbf{D}$ that is universal in the sense that for any other cocone $(C, \{\phi_j\}_{j \in \mathcal{J}})$ under $\mathbf{D}$, there exists a unique morphism $u : L \to C$ such that $u \circ \iota_j = \phi_j$ for all $j \in \mathcal{J}$.

Dually, two important examples of colimits are *coproducts* and *coequalizers*.

Definition 2.13 (Coproduct)**.** Let $\mathcal{J}$ be a discrete category with two objects. A diagram $\mathbf{D} : \mathcal{J} \to \mathcal{A}$ consists of two objects A, B. A colimit of this diagram is a *coproduct* $A + B$ together with inclusion maps $\iota_1 : A \to A + B$ and $\iota_2 : B \to A + B$.

The universal property states that for any object C with morphisms $f : A \to C$ and $g : B \to C$, there exists a unique morphism $[f, g] : A + B \to C$ such that $[f, g] \circ \iota_1 = f$ and $[f, g] \circ \iota_2 = g$. In particular, diagram (2.6) commutes:

$$\begin{tikzcd}
A \arrow[r, "\iota_1"] \arrow[dr, "f"'] & A + B \arrow[d, dashed, "{[f,g]}"] & B \arrow[l, "\iota_2"'] \arrow[dl, "g"] \\
 & C &
\end{tikzcd} \tag{2.6}$$

Many familiar structures are coproducts in various categories.

Example 2.5 (Coproducts in **Set**)**.** In the category **Set** (see definition A.1), the coproduct of sets X and Y is the disjoint union $X \sqcup Y = \bigcup_{i \in \{X,Y\}} (i \times \{i\})$ with canonical inclusions $\iota_X : X \to X \sqcup Y$ sending x to (x, X) and $\iota_Y : Y \to X \sqcup Y$ sending y to (y, Y). Given functions $f : X \to Z$ and $g : Y \to Z$, the unique mediating morphism is $[f, g] : (z, i) \mapsto f(z)$ if $i = X$ and $g(z)$ if $i = Y$.

Example 2.6 (Coproducts in **Grp**)**.** In the category **Grp** (see definition A.12), the coproduct of groups G and H is the free product $G * H$, which consists of all words in the elements of G and H modulo the relations that hold within each group. The inclusions are the canonical embeddings of G and H into the free product.

Example 2.7 (Coproducts in **Top**)**.** In the category **Top** (see definition A.13), the coproduct of topological spaces (X, τ_X) and (Y, τ_Y) is the disjoint union $X \sqcup Y$ equipped with the disjoint union topology, where a set $U \subseteq X \sqcup Y$ is open if and only if $U \cap X \in \tau_X$ and $U \cap Y \in \tau_Y$. The inclusions are continuous maps.

The dual notion to equalizers are coequalizers.

Definition 2.14 (Coequalizer)**.** Let $\mathcal{J}$ be the category with two objects and a parallel pair of morphisms between them. A diagram $\mathbf{D} : \mathcal{J} \to \mathcal{A}$ consists of two objects A, B and two morphisms $f, g : A \to B$. A colimit of this diagram is a *coequalizer* $\operatorname{coeq}(f, g)$ together with a morphism $q : B \to \operatorname{coeq}(f, g)$ such that $q \circ f = q \circ g$.

The universal property states that for any object C with a morphism $h : B \to C$ such that $h \circ f = h \circ g$, there exists a unique morphism $u : \operatorname{coeq}(f, g) \to C$ such that $u \circ q = h$. In particular, diagram (2.7) commutes:

$$\begin{tikzcd}
A \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & B \arrow[r, "q"] \arrow[rd, "h"'] & \operatorname{coeq}(f,g) \arrow[d, dashed, "u"] \\
& & C
\end{tikzcd} \tag{2.7}$$

Coequalizers generalize the notion of quotient structures in algebra. Examples of coequalizers, however, will be discussed in more detail in section 4.1, as they play a crucial role in charactering monadicity.

While limits and colimits are dual concepts thus in general different, they can coincide in certain categories or under specific conditions. The following result demonstrates this subtlety, that is, for finite families of R-modules, products (direct products) and coproducts (direct sums) coincide, while for infinite families, they generally differ.

Example 2.8 (Product and Coproduct in R-**Mod**)**.** Let R be a ring with 1 and consider the category R-**Mod** of R-modules (see definition A.14). For a finite family of R-modules $\{M_i\}_{i\in I}$ indexed by a finite set I, the direct product $\prod_{i\in I} M_i$ and the direct sum $\bigoplus_{i\in I} M_i$ are isomorphic. For an infinite family, they generally differ.

Proof. For a finite index set I, the direct sum $\bigoplus_{i\in I} M_i$ is defined as the submodule of $\prod_{i\in I} M_i$ consisting of those tuples with only finitely many nonzero components. Since I is finite, every tuple has only finitely many nonzero components, so $\bigoplus_{i\in I} M_i = \prod_{i\in I} M_i$. The canonical inclusions and projections witness this isomorphism.

For an infinite index set, consider the direct product $\prod_{i\in\mathbb{N}} R$ and direct sum $\bigoplus_{i\in\mathbb{N}} R$ where R is a nonzero ring. The element $(1,1,1,\ldots) \in \prod_{i\in\mathbb{N}} R$ has all entries nonzero, so it does not belong to $\bigoplus_{i\in\mathbb{N}} R$. Hence $\prod_{i\in\mathbb{N}} R \ncong \bigoplus_{i\in\mathbb{N}} R$. □

2.3 Adjunctions

An adjunction is the second best thing we can know about two categories other than being equivalent. In the literature, there are multiple definitions of adjunctions. Some sources define adjunctions in such a way and use other definitions as propositions. We will provide three definitions of adjunctions and prove their equivalence.

Definition 2.15 (Adjunctions via unit-counit)**.** An *adjunction* between categories $\mathcal{A}$ and $\mathcal{B}$ consists of

(1) A pair of functors $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ and $\mathbf{G} : \mathcal{B} \to \mathcal{A}$

(2) Natural transformations $\eta : \mathbf{id}_{\mathcal{A}} \Rightarrow \mathbf{GF}$ (the *unit*) and $\varepsilon : \mathbf{FG} \Rightarrow \mathbf{id}_{\mathcal{B}}$ (the *counit*)

such that the following *triangle identities* in diagram (2.8) and (2.9) hold:

$$\begin{tikzcd}
\mathbf{F} \arrow[r, Rightarrow, "\mathbf{F}\eta"] \arrow[dr, Rightarrow, "\mathbf{id_F}"'] & \mathbf{FGF} \arrow[d, Rightarrow, "\varepsilon\mathbf{F}"] \\
& \mathbf{F}
\end{tikzcd} \tag{2.8}$$

$$\begin{tikzcd}
\mathbf{G} \arrow[r, Rightarrow, "\eta\mathbf{G}"] \arrow[dr, Rightarrow, "\mathbf{id_G}"'] & \mathbf{GFG} \arrow[d, Rightarrow, "\mathbf{G}\varepsilon"] \\
& \mathbf{G}
\end{tikzcd} \tag{2.9}$$

We write $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ and say that $\mathbf{F}$ is *left adjoint* to $\mathbf{G}$ and $\mathbf{G}$ is *right adjoint* to $\mathbf{F}$. When $\mathcal{A}$ and $\mathcal{B}$ are clear from context, we write $\mathbf{F} \dashv \mathbf{G}$.

Before exploring the formal structure of adjunctions, we introduce some terminology that will help illustrate the concept. A *forgetful functor* is a functor defined by *forgetting* some structure. For example, the forgetful functor in example 2.1 from **Grp** to **Set** forgets the group structure, remembering only the underlying set. Although the term has no precise definition in the literature, it is used whenever a functor is obviously defined by forgetting structure. When $\mathbf{U} : \mathcal{B} \to \mathcal{A}$ is a forgetful functor and X is an object in $\mathcal{A}$, a *free $\mathcal{B}$-object on X with respect to* $\mathbf{U}$ is an object $\mathbf{F}X$ (up to isomorphism) that satisfies a universal property, and $\mathbf{F}$ is called the *free functor*.

Example 2.9 (Free-Forgetful Adjunction between **Set** and **Grp**)**.** The *free-forgetful adjunction* between **Set** and **Grp** consists of the functors $\mathbf{F} : \mathbf{Set} \to \mathbf{Grp}$ and $\mathbf{U} : \mathbf{Grp} \to \mathbf{Set}$ and the natural transformations $\eta_X : X \to \mathbf{UF}X$ and $\varepsilon_G : \mathbf{FU}G \to G$ such that diagrams in Definition 2.15 commute.

The unit η_X sends each element $x \in X$ to the corresponding generator in the free group $\mathbf{F}X$. The counit $\varepsilon_{\mathbf{G}}$ sends each word in the free group on the underlying set of $\mathbf{G}$ to its evaluation in $\mathbf{G}$.

There are two alternative but equivalent characterizations of adjunctions that will be useful in different contexts.

Definition 2.16 (Adjunctions via Hom-set)**.** An *adjunction* between categories $\mathcal{A}$ and $\mathcal{B}$ consists of

(1) A pair of functors $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ and $\mathbf{G} : \mathcal{B} \to \mathcal{A}$

(2) A natural isomorphism $\lfloor - \rfloor : \mathcal{B}(\mathbf{F}-, -) \to \mathcal{A}(-, \mathbf{G}-)$ with inverse $\lceil - \rceil : \mathcal{A}(-, \mathbf{G}-) \to \mathcal{B}(\mathbf{F}-, -)$

That is, for each object $A \in \mathrm{Ob}\,\mathcal{A}$ and $B \in \mathrm{Ob}\,\mathcal{B}$, there is a bijection:

$$\lfloor - \rfloor_{A,B} : \mathcal{B}(\mathbf{F}A, B) \to \mathcal{A}(A, \mathbf{G}B)$$

such that for any morphisms $f : A' \to A$ in $\mathcal{A}$ and $g : B \to B'$ in $\mathcal{B}$, diagram (2.10) commutes:

$$\begin{tikzcd}
\mathcal{B}(\mathbf{F}A, B) \arrow[r, "\lfloor - \rfloor_{A,B}"] \arrow[d, "g \circ (-) \circ \mathbf{F}f"'] & \mathcal{A}(A, \mathbf{G}B) \arrow[d, "\mathbf{G}g \circ (-) \circ f"] \\
\mathcal{B}(\mathbf{F}A', B') \arrow[r, "\lfloor - \rfloor_{A',B'}"] & \mathcal{A}(A', \mathbf{G}B')
\end{tikzcd} \tag{2.10}$$

Definition 2.17 (Adjunctions via universal property)**.** An *adjunction* between categories $\mathcal{A}$ and $\mathcal{B}$ consists of

(1) A functor $\mathbf{G} : \mathcal{B} \to \mathcal{A}$

(2) For each object $A \in \mathrm{Ob}\,\mathcal{A}$, an object $\mathbf{F}A \in \mathrm{Ob}\,\mathcal{B}$ and a morphism $\eta_A : A \to \mathbf{GF}A$

such that for each object $A \in \mathrm{Ob}\,\mathcal{A}$ and each morphism $f : A \to \mathbf{G}B$ in $\mathcal{A}$, there exists a unique morphism $\overline{f} : \mathbf{F}A \to B$ in $\mathcal{B}$ making diagram (2.11) commute:

$$\begin{tikzcd}
A \arrow[r, "\eta_A"] \arrow[dr, "f"'] & \mathbf{GF}A \arrow[d, "\mathbf{G}\overline{f}"] \\
& \mathbf{G}B
\end{tikzcd} \tag{2.11}$$

These three definitions offer different perspectives on adjunctions: the unit-counit definition emphasizes the natural transformations, the hom-set definition reveals the duality between left and right adjoints, and the universal property definition connects to universal constructions. We now establish their equivalence.

Theorem 2.1 (Equivalence of Adjunction Definitions)**.** *The three definitions of adjunction given above are equivalent.*

Proof. We will show the implications in a cycle: Definition 2.15 ⇒ Definition 2.16 ⇒ Definition 2.17 ⇒ Definition 2.15.

Definition 2.15 ⇒ Definition 2.16. Given the unit-counit definition, we define the natural isomorphism $\lfloor - \rfloor$ by:

$$\lfloor f \rfloor = \mathbf{G}f \circ \eta_A$$

for $f : \mathbf{F}A \to B$. The inverse is given by:

$$\lceil g \rceil = \varepsilon_B \circ \mathbf{F}g$$

for $g : A \to \mathbf{G}B$.

To verify naturality, consider morphisms $h : A' \to A$ and $k : B \to B'$. We need to show:

$$\lfloor k \circ f \circ \mathbf{F}h \rfloor = \mathbf{G}k \circ \lfloor f \rfloor \circ h$$

This follows from the naturality of η and the functoriality of G:

$$\begin{aligned} \lfloor k \circ f \circ \mathbf{F}h \rfloor &= \mathbf{G}k \circ f \circ \mathbf{F}h \circ \eta_{A'} \\ &= \mathbf{G}k \circ \mathbf{G}f \circ \mathbf{GF}h \circ \eta_{A'} \\ &= \mathbf{G}k \circ \mathbf{G}f \circ \eta_A \circ h \\ &= \mathbf{G}k \circ \lfloor f \rfloor \circ h \end{aligned}$$

The triangle identities ensure that $\lfloor - \rfloor$ and $\lceil - \rceil$ are indeed inverses.

Definition 2.16 ⇒ Definition 2.17. Given the natural isomorphism $\lfloor - \rfloor$, we define the unit $\eta_A : A \to \mathbf{GF}A$ as:

$$\eta_A = \lfloor \mathbf{id}_{\mathbf{F}A} \rfloor$$

The universal property follows from the naturality of $\lfloor - \rfloor$. Given $f : A \to \mathbf{G}B$, we define $\overline{f} = \lceil f \rceil$. The commutativity of the diagram follows from the naturality of $\lfloor - \rfloor$.

Definition 2.17 ⇒ Definition 2.15. Given the universal property, we can extend $\mathbf{F}$ to a functor by defining $\mathbf{F}f = \overline{\mathbf{G}f \circ \eta_A}$ for $f : A \to A'$.

We define the counit $\varepsilon_B : \mathbf{FG}B \to B$ as $\varepsilon_B = \overline{\mathbf{id}_{\mathbf{G}B}}$. The triangle identities follow from the universal property and the uniqueness of the mediating morphisms. □

We explicitly state the Hom-set formulation of adjunctions in terms of the unit and counit from Theorem 2.1 as a proposition.

Proposition 2.1 (Adjoint transposition)**.** *Let $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ be an adjunction with unit η and counit ε. Then for every $A \in \mathrm{Ob}\,(\mathcal{A})$ and $B \in \mathrm{Ob}\,(\mathcal{B})$ there is a natural bijection*

$$\lfloor - \rfloor : \mathcal{B}(\mathbf{F}A, B) \cong \mathcal{A}(A, \mathbf{G}B) : \lceil - \rceil$$

sending a morphism $h : A \to \mathbf{G}B$ to its adjunct

$$\lceil h \rceil := \varepsilon_B \circ \mathbf{F}h : \mathbf{F}A \to B,$$

with inverse given by

$$\lfloor h \rfloor := \mathbf{G}h \circ \eta_A : A \to \mathbf{G}B.$$

These assignments are mutually inverse.

Adjunctions capture the idea of *optimal approximation* between categories, they are weaker than equivalences, but still have strong structural implications and well-behaved properties. A fundamental result about adjoint functors is their uniqueness up to natural isomorphism.

Lemma 2.1 (Uniqueness of adjoint functors)**.** *If $\mathbf{F}$ and $\mathbf{F}'$ are both left (right) adjoints to $\mathbf{G} : \mathcal{B} \to \mathcal{A}$, then $\mathbf{F} \cong \mathbf{F}'$ naturally.*

Proof Sketch. By the Yoneda Lemma[1], the natural isomorphism

$$\mathcal{B}(\mathbf{F}-, -) \cong \mathcal{A}(-\mathbf{G}-) \cong \mathcal{B}(\mathbf{F}'-, -)$$

induces a natural isomorphism $\mathbf{F} \cong \mathbf{F}'$. The case for right adjoints is similar. □

Adjunctions can also be composed to form new adjunctions, just like equivalences.

Lemma 2.2 (Composition of Adjunctions)**.** *Given adjunctions $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ and $\mathbf{F}' \dashv \mathbf{G}' : \mathcal{B} \to \mathcal{C}$*

$$\begin{tikzcd}
\mathcal{A} \arrow[r, shift left, "\mathbf{F}"] \arrow[r, phantom, "\perp"] & \mathcal{B} \arrow[l, shift left, "\mathbf{G}"] \arrow[r, shift left, "\mathbf{F}'"] \arrow[r, phantom, "\perp"] & \mathcal{C} \arrow[l, shift left, "\mathbf{G}'"] \arrow[r, phantom, "\rightsquigarrow"] & \mathcal{A} \arrow[r, shift left, "\mathbf{F}' \circ \mathbf{F}"] \arrow[r, phantom, "\perp"] & \mathcal{C} \arrow[l, shift left, "\mathbf{G} \circ \mathbf{G}'"]
\end{tikzcd}$$

then the composition $\mathbf{F}' \circ \mathbf{F} \dashv \mathbf{G} \circ \mathbf{G}' : \mathcal{A} \to \mathcal{C}$ is an adjunction.

[1] Yoneda Lemma is beyond the scope of this thesis, so we are only providing a sketch of proof.

Proof. There are natural isomorphisms $\mathcal{C}(\mathbf{F}'\mathbf{F}x, y) \cong \mathcal{B}(\mathbf{F}x, \mathbf{G}'y) \cong \mathcal{A}(x, \mathbf{G}\mathbf{G}'y)$, the first defined using $\mathbf{F}'$ and $\mathbf{G}'$ and the second defined using $\mathbf{F}$ and $\mathbf{G}$. □

All properties are preserved under an isomorphism of categories, most properties are also preserved under an equivalence of categories. Since adjunctions are weaker than equivalences, what are the properties preserved by adjunctions? It turns out adjunctions exhibit important preservation properties with respect to limits and colimits.

Theorem 2.2 (Right Adjoints Preserve Limits)**.** *If* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$*, then* $\mathbf{G}$ *preserves all limits that exist in* $\mathcal{B}$*.*

Proof. Let $(L, \{\pi_j\})$ be a limit of a diagram $\mathbf{D} : \mathcal{J} \to \mathcal{B}$. We need to show that $(\mathbf{G}L, \{\mathbf{G}\pi_j\})$ is a limit of $\mathbf{G} \circ \mathbf{D} : \mathcal{J} \to \mathcal{A}$. Given any cone $(C, \{\phi_j\})$ over $\mathbf{G} \circ \mathbf{D}$, by the adjunction, each $\phi_j : C \to \mathbf{G}D_j$ corresponds to a unique morphism $\psi_j : \mathbf{F}C \to \mathbf{D}_j$. The collection $\{\psi_j\}$ forms a cone over $\mathbf{D}$, so there exists a unique morphism $u : \mathbf{F}C \to L$ such that $\pi_j \circ u = \psi_j$ for all j. By the adjunction again, u corresponds to a unique morphism $\overline{u} : C \to \mathbf{G}L$ such that $\mathbf{G}\pi_j \circ \overline{u} = \phi_j$ for all j. This establishes the universal property of the limit. □

By duality, left adjoints preserve colimits.

Theorem 2.3 (Left Adjoints Preserve Colimits)**.** *If* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$*, then* $\mathbf{F}$ *preserves all colimits that exist in* $\mathcal{A}$*.*

We conclude this chapter with some notable examples of adjunctions that illustrate their ubiquity and utility in various mathematical contexts.

Example 2.10 (Product-Hom Adjunction)**.** In the category **Set**, for any set A, we have an adjunction[2]:

$$(-) \times A \dashv \mathbf{Set}(A, -) : \mathbf{Set} \to \mathbf{Set}$$

This adjunction embodies the mathematical formalization of currying. The natural isomorphism states that a function $f : X \times A \to Y$ from the cartesian product corresponds uniquely to a function $\overline{f} : X \to \mathbf{Set}(A, Y)$ that takes an element of X and returns a function from A to Y:

$$\mathbf{Set}(X \times A, Y) \cong \mathbf{Set}(X, \mathbf{Set}(A, Y))$$

[2] Being Cartesian-closed should be enough.

Explicitly, for $f : X \times A \to Y$, we define $\overline{f}(x)(a) = f(x, a)$. Conversely, given $g : X \to \mathbf{Set}(A, Y)$, we define $\widehat{g}(x, a) = g(x)(a)$. This is exactly currying: transforming a function of two arguments into a function that returns another function. The unit $\eta_X : X \to \mathbf{Set}(A, X \times A)$ sends x to the function $a \mapsto (x, a)$, while the counit $\varepsilon_Y : \mathbf{Set}(A, Y) \times A \to Y$ is the evaluation map $(f, a) \mapsto f(a)$.

In fact, for readers familiar with functional programming, example 2.10 captures the essence of how functions are treated in languages like Haskell or ML, where functions are inherently *curried*.

```
-- currying
curry :: ((a, b) -> c) -> (a -> b -> c)
curry f x y = f (x, y)

-- uncurrying
uncurry :: (a -> b -> c) -> ((a, b) -> c)
uncurry f (x, y) = f x y
```

This principle of a natural Hom-set isomorphism definition 2.16 is not limited to **Set**. It finds a classical formulation in order theory, where the adjunction manifests as a Galois connection.

Example 2.11 (Galois Connections as Adjunctions)**.** A *Galois connection* between partially ordered sets $(P, \leq_P)$ and $(Q, \leq_Q)$ consists of two monotone functions $f : P \to Q$ and $g : Q \to P$ such that for all $p \in P$ and $q \in Q$:

$$f(p) \leq_Q q \iff p \leq_P g(q)$$

We say that f is the *lower adjoint* and g is the *upper adjoint*.

In fact, a Galois connection is precisely an adjunction in the category **Pos** of posets. Specifically, viewing posets as categories where there is a unique morphism $p \to q$ if and only if $p \leq q$, the condition

$$f(p) \leq q \iff p \leq g(q)$$

becomes the natural bijection of hom-sets:

$$Q(f(p), q) \cong P(p, g(q))$$

In programming language semantics, type inference and program analysis often give rise to Galois connections. For instance, consider a static analysis that tracks abstract values in a program. If P is the poset of concrete program states and Q is the poset of abstract values, then:

> The *abstraction function* $\alpha : P \to Q$ sends concrete states to their abstract counterparts. α is monotone because more precise concrete states map to more precise abstract values.
>
> The *concretization function* $\gamma : Q \to P$ sends abstract values to the set of all concrete states they represent. γ is also monotone, as more precise abstract values represent smaller sets of concrete states.

The Galois connection $\alpha \dashv \gamma$ ensures *soundness*: a concrete property holds in all states represented by an abstract value exactly when the corresponding abstract property holds. This is expressed by:

$$\alpha(c) \leq_Q a \iff c \leq_P \gamma(a)$$

where $c \in P$ is a concrete state and $a \in Q$ is an abstract value.

Perhaps the most foundational adjunction in abstract algebra is the one relating tensor products and Hom-functors. This example is particularly instructive as it directly illustrates the equivalence between the Hom-set definition of an adjunction (Definition 2.16) and the universal property definition (Definition 2.17).

Example 2.12 (Tensor-Hom Adjunction)**.** Let R be a commutative ring. In the category R-**Mod**of R-modules, for a fixed R-module A, we have the adjunction:

$$A \otimes_R (-) \dashv \mathrm{Hom}_R(A, -) : R\text{-}\mathbf{Mod} \to R\text{-}\mathbf{Mod}$$

The functor $A \otimes_R (-)$ sends an R-module N to the tensor product $A \otimes_R N$, and a morphism $f : N \to M$ to $\mathrm{id}_A \otimes_R f : A \otimes_R N \to A \otimes_R M$. The functor $\mathrm{Hom}_R(A, -)$ sends an R-module N to the abelian group $\mathrm{Hom}_R(A, N)$ of R-module homomorphisms from A to N, and a morphism $f : N \to M$ to the post-composition map $f \circ (-)$.

The unit

$$\eta_N : N \to \mathrm{Hom}_R(A, A \otimes_R N)$$

is defined by sending $n \in N$ to the homomorphism $a \mapsto a \otimes n$. The counit $\varepsilon_M : A \otimes_R \mathrm{Hom}_R(A, M) \to M$ is the evaluation map $a \otimes f \mapsto f(a)$. The natural isomorphism

$$\lfloor - \rfloor_{N,M} : \mathrm{Hom}_R(A \otimes_R N, M) \to \mathrm{Hom}_R(N, \mathrm{Hom}_R(A, M))$$

is given by currying: for $g : A \otimes_R N \to M$, define $\lfloor g \rfloor : N \to \mathrm{Hom}_R(A, M)$ by $n \mapsto (a \mapsto g(a \otimes n))$. Its inverse sends $h : N \to \mathrm{Hom}_R(A, M)$ to $\lceil h \rceil : A \otimes_R N \to M$ defined by $a \otimes n \mapsto h(n)(a)$.

This adjunction directly demonstrates the equivalence between the Hom-set definition (Definition 2.16) and the universal property definition (Definition 2.17).

The *universal property* of the tensor product $A \otimes_R N$ states that it is the universal object for R-bilinear maps out of $A \times N$. That is, for any R-module M, there is a natural bijection between the set of R-linear maps (from the tensor product) and the set of R-bilinear maps (from the product):

$$\mathrm{Hom}_R(A \otimes_R N, M) \cong \text{Bilin}_R(A, N; M)$$

This is, in effect, an instance of Definition 2.20 (Adjunction via universal property), where $A \otimes_R (-)$ is the free construction. Separately, the familiar process of "currying" provides its own natural bijection between R-bilinear maps and iterated R-linear maps:

$$\text{Bilin}_R(A, N; M) \cong \mathrm{Hom}_R(N, \mathrm{Hom}_R(A, M))$$

An element g on the right ($g : N \to \mathrm{Hom}_R(A, M)$) corresponds to the bilinear map $\varphi(a, n) = g(n)(a)$ on the left. By composing these two bijections, we recover the Hom-set adjunction from definition 2.16:

$$\mathrm{Hom}_R(A \otimes_R N, M) \cong \mathrm{Hom}_R(N, \mathrm{Hom}_R(A, M))$$

This shows how the universal property of the tensor product *is* the Hom-set adjunction. This equivalence—between a universal construction (Definition 2.17) and a Hom-set bijection (Definition 2.16)—is a common pattern. It highlights the unit-counit definition (Definition 2.15) as a distinct, third perspective, which will be central to defining monads in the next chapter.

Chapter 3

Monads and Algebras

Monads provide a unifying framework for describing algebraic and computational structure within category theory. They arise naturally when one studies constructions that combine data with additional structure, such as context, effects, or notions of composition, while remaining compatible with categorical composition. From this perspective, a monad encapsulates a way of building complex objects from simpler ones in a coherent and principled manner.

The notion of a monad was introduced by Godement [God58] in the study of algebraic topology, and later appeared under various names, including the "dual standard construction", "triple", "monoid", and "triad" [ML98]. In computer science, monads became influential through Moggi's work on the computational λ calculus [Mog88], and were later popularized by Wadler's article *Monads for Functional Programming* [Wad95]. A concise categorical characterization is often summarized by the slogan that *monads are monoids in the category of endofunctors*, although this description is most meaningful after the basic structure of monads has been developed.

In this chapter, we introduce the fundamental theory of monads in category theory. We begin with the definition of monads and their algebras in section 3.1, and then discuss free algebras in section 3.2. We conclude by presenting the two canonical categorical constructions associated with a monad, namely the Kleisli category and the Eilenberg Moore category, which provide complementary perspectives on monadic structure in section 3.3.

3.1 Monads and Their Algebras

In this section, we introduce the notion of monads and their algebras.

Definition 3.1 (Monad)**.** A *monad* on a category $\mathcal{A}$ is a triple $\mathbb{T} = (\mathbf{T}, \eta, \mu)$ where:

1. $\mathbf{T} : \mathcal{A} \to \mathcal{A}$ is an endofunctor [1]
2. $\eta : \mathbf{id}_{\mathcal{A}} \Rightarrow \mathbf{T}$ is a natural transformation (the *unit*)
3. $\mu : \mathbf{TT} \Rightarrow \mathbf{T}$ is a natural transformation (the *multiplication*)

such that diagrams (3.1), (3.2), and (3.3) commute:

$$\begin{tikzcd} \mathbf{T} \arrow[r, Rightarrow, "\mathbf{T}\eta"] \arrow[dr, Rightarrow, "\mathbf{id_T}"'] & \mathbf{TT} \arrow[d, Rightarrow, "\mu"] \\ & \mathbf{T} \end{tikzcd} \quad (3.1) \qquad \begin{tikzcd} \mathbf{T} \arrow[r, Rightarrow, "\eta\mathbf{T}"] \arrow[dr, Rightarrow, "\mathbf{id_T}"'] & \mathbf{TT} \arrow[d, Rightarrow, "\mu"] \\ & \mathbf{T} \end{tikzcd} \quad (3.2) \qquad \begin{tikzcd} \mathbf{TTT} \arrow[r, Rightarrow, "\mathbf{T}\mu"] \arrow[d, Rightarrow, "\mu\mathbf{T}"'] & \mathbf{TT} \arrow[d, Rightarrow, "\mu"] \\ \mathbf{TT} \arrow[r, Rightarrow, "\mu"'] & \mathbf{T} \end{tikzcd} \quad (3.3)$$

They are called the *left unit*, *right unit*, and *associativity* axioms respectively.

Intuitively, the monad axioms ensure that the unit and multiplication behave like the unit and multiplication of a monoid, and that is why monads are sometimes referred to as *monoids in the category of endofunctors*.

Example 3.1 (Maybe Monad)**.** The *maybe monad* on Set is defined by:

- $\mathbf{T}X = X \sqcup \{\bot\}$ where $\bot$ represents "nothing".
- $\eta_X : X \hookrightarrow X \sqcup \{\bot\}$ is the inclusion map.
- $\mu_X : (X \sqcup \{\bot\}) \sqcup \{\bot\} \to X \sqcup \{\bot\}$

$$\mu_X : \begin{cases} x \mapsto x \\ \bot \mapsto \bot \end{cases}$$

The monad laws are easily verified.

One of the most common ways in which monads arise is from adjunctions. In particular, every adjunction gives rise to a monad.

[1] If η and μ is clear from context, we refer to monad $\mathbb{T}$ simply by its underlying endofunctor $\mathbf{T}$.

Theorem 3.1 (Every Adjunction Determines a Monad)**.** *Let* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ *be an adjunction with unit* $\eta : \mathbf{id}_{\mathcal{A}} \Rightarrow \mathbf{GF}$ *and counit* $\varepsilon : \mathbf{FG} \Rightarrow \mathbf{id}_{\mathcal{B}}$*. Then* $(\mathbf{GF}, \eta, \mathbf{G}\varepsilon\mathbf{F})$ *is a monad on* $\mathcal{A}$*.*

Proof. The proof is a straightforward verification of the monad laws. Left compose $\mathbf{G}$ on the entire diagram (3.1) gives diagram (3.4), and right compose $\mathbf{F}$ on diagram (2.9) gives diagram (3.5), proving the left and right unit laws respectively.

$$\begin{tikzcd}
\mathbf{GF} \arrow[r, Rightarrow, "\mathbf{GF}\eta"] \arrow[dr, Rightarrow, "\mathbf{id_{GF}}"'] & \mathbf{GFGF} \arrow[d, Rightarrow, "\mathbf{G}\varepsilon\mathbf{F}"] \\
& \mathbf{GF}
\end{tikzcd} \tag{3.4}$$

$$\begin{tikzcd}
\mathbf{GF} \arrow[r, Rightarrow, "\eta\mathbf{GF}"] \arrow[dr, Rightarrow, "\mathbf{id_{GF}}"'] & \mathbf{GFGF} \arrow[d, Rightarrow, "\mathbf{G}\varepsilon\mathbf{F}"] \\
& \mathbf{GF}
\end{tikzcd} \tag{3.5}$$

The associativity law follows from naturality of ε as shown in diagram (3.6).

$$\begin{tikzcd}
\mathbf{GFGFGF} \arrow[r, Rightarrow, "\mathbf{GFG}\varepsilon\mathbf{F}"] \arrow[d, Rightarrow, "\mathbf{G}\varepsilon\mathbf{F}"'] & \mathbf{GFGF} \arrow[d, Rightarrow, "\mathbf{G}\varepsilon\mathbf{F}"] \\
\mathbf{GFGF} \arrow[r, Rightarrow, "\mathbf{G}\varepsilon\mathbf{F}"'] & \mathbf{GF}
\end{tikzcd} \tag{3.6}$$

□

When monads were first discovered in the 1950's and it was found that adjunctions give rise to monads, a natural and fundamental question was raised by Hilton et al. [Mac63]: Whether every monad comes from an adjunction? The answer is yes, and two well-known solutions appeared, which leads us to the construction of two special categories associated with any monad: the Eilenberg-Moore category [EM65] and the Kleisli category [Kle65]. Both solutions, however, are deeply related to the concept of *algebras* for a monad. To see this, we first define algebras for endofunctors.

Definition 3.2 (Functor Algebra)**.** Let $\mathbf{F}$ be a endofunctor on category $\mathcal{A}$. A *algebra* of $\mathbf{F}$ is a pair (A, α) where A is an object of $\mathcal{A}$ (called the *carrier*) and $\alpha : \mathbf{F}A \to A$ is a morphism in $\mathcal{A}$ (called the *algebra structure*). A *homomorphism* between two algebras (A, α) to (B, β) is a morphism $f : A \to B$ in $\mathcal{A}$ such that diagram (3.7) commutes:

$$\begin{tikzcd}
\mathbf{F}A \arrow[r, "\mathbf{F}f"] \arrow[d, "\alpha"'] & \mathbf{F}B \arrow[d, "\beta"] \\
A \arrow[r, "f"'] & B
\end{tikzcd} \tag{3.7}$$

All $\mathbf{F}$-algebras and their morphisms form a category, denoted $\mathbf{F}$-$\mathbf{Alg}$, whose objects are pairs $(A, \alpha : \mathbf{F}A \to A)$ and whose morphisms are $\mathbf{F}$-algebra morphisms. Compositions and identities are inherited from $\mathcal{A}$.

Interestingly, the initial objects in $\mathbf{F}$-$\mathbf{Alg}$, called an *initial* $\mathbf{F}$*-algebra*, often have significant computational interpretations, such as representing inductive data types, as detailed in lemma 3.1.

Lemma 3.1 (Lambek)**.** *If* $(\mu\mathbf{F}, \alpha : \mathbf{F}\mu\mathbf{F} \to \mu\mathbf{F})$ *is an initial* $\mathbf{F}$*-algebra, then* α *is an isomorphism.*

Proof. Since $(\mu\mathbf{F}, \alpha)$ is an initial $\mathbf{F}$-algebra, there exists a unique $\mathbf{F}$-algebra morphism $\beta : \mu\mathbf{F} \to \mathbf{F}\mu\mathbf{F}$ such that the following diagram commutes:

$$\begin{tikzcd}
\mathbf{F}\mu\mathbf{F} \arrow[r, "\mathbf{F}\beta"] \arrow[d, "\alpha"'] & \mathbf{F}\mathbf{F}\mu\mathbf{F} \arrow[d, "\mathbf{F}\alpha"] \\
\mu\mathbf{F} \arrow[r, "\beta"'] & \mathbf{F}\mu\mathbf{F}
\end{tikzcd}$$

(1) $\alpha \circ \beta = \mathbf{id}_{\mu\mathbf{F}}$. First note that $\alpha \circ \beta$ is an $\mathbf{F}$-algebra homomorphism from $(\mu\mathbf{F}, \alpha)$ to itself, because

$$\begin{aligned}
(\alpha \circ \beta) \circ \alpha &= \alpha \circ (\beta \circ \alpha) \\
&= \alpha \circ (\mathbf{F}\alpha \circ \mathbf{F}\beta) \\
&= \alpha \circ \mathbf{F}\alpha \circ \mathbf{F}\beta \\
&= \alpha \circ \mathbf{F}(\alpha \circ \beta)
\end{aligned}$$

But $(\mu\mathbf{F}, \alpha)$ is initial, so there is only one algebra morphism $(\mu\mathbf{F}, \alpha) \to (\mu\mathbf{F}, \alpha)$. The identity $\mathrm{id}_{\mu\mathbf{F}}$ is certainly such a morphism, therefore

$$\alpha \circ \beta = \mathrm{id}_{\mu\mathbf{F}} \,.$$

(2) $\beta \circ \alpha = \mathrm{id}_{\mathbf{F}\mu\mathbf{F}}$. We already have

$$\beta \circ \alpha = \mathbf{F}\alpha \circ \mathbf{F}\beta.$$

Apply the functor $\mathbf{F}$ to the equality $\alpha \circ \beta = \mathrm{id}_{\mu\mathbf{F}}$ gives us

$$\begin{aligned}\mathbf{F}\alpha \circ \mathbf{F}\beta &= \mathbf{F}(\alpha \circ \beta) \\ &= \mathbf{F}\mathrm{id}_{\mu\mathbf{F}} \\ &= \mathrm{id}_{\mathbf{F}\mu\mathbf{F}}\,.\end{aligned}$$

i.e.,

$$\beta \circ \alpha = \mathrm{id}_{\mathbf{F}\mu\mathbf{F}}\,.$$

So α and β are mutual inverses, hence α is an isomorphism, as claimed.

□

Analogous to viewing groups as sets with structure, we can view monads as special endofunctors with structures (in this case, η and μ). So *monad algebras* are just *functor algebras* for the underlying endofunctor of the monad, satisfying additional conditions regarding the unit and multiplication of the monad.

Definition 3.3 (Monad Algebra)**.** Let $\mathbb{T} = (\mathbf{T}, \eta, \mu)$ be a monad on a category $\mathcal{A}$. A $\mathbb{T}$-*algebra* is a pair (A, α) where A is an object of $\mathcal{A}$, and $\alpha : \mathbf{T}A \to A$ is a morphism in $\mathcal{A}$ (called the *algebra structure*), such that diagram (3.8) (unit property) and (3.9) (action property) commute:

$$\begin{tikzcd} A \arrow[r, "\eta_A"] \arrow[dr, "\mathbf{id}_A"'] & \mathbf{T}A \arrow[d, "\alpha"] \\ & A \end{tikzcd} \qquad (3.8)$$

$$\begin{tikzcd} \mathbf{T}\mathbf{T}A \arrow[r, "\mathbf{T}\alpha"] \arrow[d, "\mu_A"'] & \mathbf{T}A \arrow[d, "\alpha"] \\ \mathbf{T}A \arrow[r, "\alpha"'] & A \end{tikzcd} \qquad (3.9)$$

$\mathbb{T}$-algebra homomorphisms are just $\mathbf{T}$-algebra homomorphisms defined in definition 3.2. Similarly, when it is clear from context whether we are referring to the monad or its underlying endofunctor, we write $\mathbf{T}$-algebra instead of $\mathbb{T}$-algebra to denote a monad algebra.

In 1965, Samuel Eilenberg and John C. Moore introduced the construction of the category of algebras for a monad [EM65], which was called a *triple* at that time. All algebras of a monad $\mathbf{T}$ and their morphisms form a category called the Eilenberg-Moore category of $\mathbf{T}$. This category, together with a canonical adjunction, provides one solution to the question raised by Hilton et al.

Definition 3.4 (Eilenberg-Moore Category)**.** Let $(\mathbf{T}, \eta, \mu)$ be a monad on a category $\mathcal{A}$. The *Eilenberg-Moore category* $\mathcal{A}^{\mathbf{T}}$ is the category where objects are $\mathbf{T}$-algebras and morphisms are $\mathbf{T}$-algebra morphisms. Compositions are inherited from $\mathcal{A}$.

Theorem 3.2 (Every monad is defined by its $\mathbf{T}$-algebras [ML98])**.** *Let $(\mathbf{T}, \eta, \mu)$ be a monad on a category $\mathcal{A}$. There's an adjunction $\mathbf{F}^{\mathbf{T}} \dashv \mathbf{G}^{\mathbf{T}} : \mathcal{A} \to \mathcal{A}^{\mathbf{T}}$ given by assignments (3.10) and (3.11) respectively such that $\mathbf{T} = (\mathbf{G}^{\mathbf{T}} \circ \mathbf{F}^{\mathbf{T}}, \eta^{\mathbf{T}}, \mathbf{G}^{\mathbf{T}}\varepsilon^{\mathbf{T}}\mathbf{F}^{\mathbf{T}})$, where $\eta^{\mathbf{T}}$ and $\varepsilon^{\mathbf{T}}$ are the unit and counit of the adjunction, respectively.*

$$\begin{tikzcd}
X \arrow[d, "f"'] & & (\mathbf{T}X, \mu_X) \arrow[d, "\mathbf{T}f"] \\
Y \arrow[r, phantom, "\overset{\mathbf{F}^{\mathbf{T}}}{\longmapsto}"] & & (\mathbf{T}Y, \mu_Y).
\end{tikzcd} \tag{3.10}$$

$$\begin{tikzcd}
(X, \alpha) \arrow[d, "f"'] & & X \arrow[d, "f"'] \\
(Y, \beta) \arrow[r, phantom, "\overset{\mathbf{G}^{\mathbf{T}}}{\longmapsto}"] & & Y.
\end{tikzcd} \tag{3.11}$$

Proof. The functor $\mathbf{G}^{\mathbf{T}} : \mathcal{A}^{\mathbf{T}} \to \mathcal{A}$ is the evident functor which simply forgets the structure map of each $\mathbf{T}$-algebra. On the other hand, for each X in $\mathrm{Ob}\,\mathcal{A}$, the pair $(\mathbf{T}X, \mu_X : \mathbf{TT}X \to \mathbf{T}X)$ is a $\mathbf{T}$-algebra (called the *free* $\mathbf{T}$*-algebra on* X), in view of the associative and (left) unit laws for the monad $\mathbf{T}$. Hence $X \mapsto (\mathbf{T}X, \mu_X)$ does indeed define a functor $\mathbf{F}^{\mathbf{T}} : \mathcal{A} \to \mathcal{A}^{\mathbf{T}}$, as asserted.

Then $\mathbf{G}^{\mathbf{T}} \circ \mathbf{F}^{\mathbf{T}}X = \mathbf{G}^{\mathbf{T}}(\mathbf{T}X, \mu_X) = \mathbf{T}X$, so the unit of the given monad is a natural transformation $\eta = \eta^{\mathbf{T}} : \mathbf{id}_{\mathcal{A}} \Rightarrow \mathbf{G}^{\mathbf{T}} \circ \mathbf{F}^{\mathbf{T}}$.

On the other hand, $\mathbf{F}^{\mathbf{T}} \circ \mathbf{G}^{\mathbf{T}}(X, h) = \mathbf{F}^{\mathbf{T}}X = (\mathbf{T}X, \mu_X)$. The first square in the definition of a $\mathbf{T}$-algebra (X, h) states that the structure map is $h : \mathbf{T}X \to X$. The resulting transformation $\varepsilon^{\mathbf{T}}_{(X,h)} = h : \mathbf{F}^{\mathbf{T}} \circ \mathbf{G}^{\mathbf{T}}(X, h) \to (X, h)$ is natural, by the definition of a morphism of $\mathbf{T}$-algebras.

The triangular identities for an adjunction read:

$$\begin{tikzcd}
\mathbf{T}X \arrow[r, "\mathbf{T}\eta_X"] \arrow[dr, equal] & \mathbf{TT}X \arrow[d, "\mu_X"] \\
& \mathbf{T}X
\end{tikzcd} \tag{3.12}$$

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[dr, equal] & \mathbf{T}X \arrow[d, "h"] \\
& X
\end{tikzcd} \tag{3.13}$$

Diagram (3.12) holds by the (right) unit law for $\mathbf{T}$, and diagram (3.13) holds by the unit law for a $\mathbf{T}$-algebra. Therefore $\eta^{\mathbf{T}}$ and $\varepsilon^{\mathbf{T}}$ define an adjunction, as stated.

This adjunction thus determines a monad in $\mathcal{A}$. The endofunctor $\mathbf{G}^{\mathbf{T}} \circ \mathbf{F}^{\mathbf{T}}$ is the original $\mathbf{T}$, its unit $\eta^{\mathbf{T}}$ is the original unit, and its multiplication $\mu^{\mathbf{T}} = \mathbf{G}^{\mathbf{T}}\varepsilon^{\mathbf{T}}\mathbf{F}^{\mathbf{T}}$

has $\mu_X^{\mathbf{T}} = \mathbf{G}^{\mathbf{T}}\varepsilon^{\mathbf{T}}_{(\mathbf{T}X,\mu_X)} = \mathbf{G}^{\mathbf{T}}\mu_X = \mu_X$, so is the original multiplication of $\mathbf{T}$. The proof is complete. □

In theorem 3.2, we mentioned a important special class of $\mathbf{T}$-algebras called *free* $\mathbf{T}$*-algebras*. which are of special interest to this thesis. To see that free $\mathbf{T}$-algebras are well-defined $\mathbf{T}$-algebras, we now make a formal definition as follows.

Definition 3.5 (Free $\mathbf{T}$-algebra)**.** Let $(\mathbf{T}, \eta, \mu)$ be a monad on a category $\mathcal{A}$, then

$$(\mathbf{T}X, \mu_X : \mathbf{TT}X \to \mathbf{T}X)$$

is called a *free* $\mathbf{T}$*-algebra* on object X in $\mathcal{A}$.

To see that a free $\mathbf{T}$-algebra $(\mathbf{T}X, \mu_X)$ is a well-defined $\mathbf{T}$-algebra, we need to verify the two properties of a $\mathbf{T}$-algebra as shown in diagram (3.8) and (3.9). It is easily checked that the unit property and the action property follow from the right unit law (3.2) and the associativity law (3.3) of the monad respectively.

Note from definition 3.3 that $(\mathbf{T}X, \mu_X)$ is precisely the image of object X under the free functor $\mathbf{F}^{\mathbf{T}}$ defined in theorem 3.2. Since we have an adjunction $\mathbf{F}^{\mathbf{T}} \dashv \mathbf{G}^{\mathbf{T}} : \mathcal{A} \to \mathcal{A}^{\mathbf{T}}$, by the definition 2.17 of adjunctions via universal property, free $\mathbf{T}$-algebras are indeed free in the categorical sense, as stated in lemma 3.2.

Lemma 3.2 (Universal Property of Free $\mathbf{T}$-algebras)**.** *Let $(\mathbf{T}, \eta, \mu)$ be a monad on a category $\mathcal{A}$. For each object A in* $\mathrm{Ob}\,\mathcal{A}$*, and each morphism $f : A \to \mathbf{T}B$ in $\mathcal{A}$, there exists a unique morphism of $\mathbf{T}$-algebras $\overline{f} : (\mathbf{T}A, \mu_A) \to (\mathbf{T}B, \mu_B)$ such that the following diagram commutes:*

$$\begin{tikzcd}
A \arrow[r, "\eta_A^{\mathbf{T}}"] \arrow[dr, "f"'] & \mathbf{T}A \arrow[d, dashed, "\mathbf{U}\overline{f}"] \\
 & \mathbf{T}B
\end{tikzcd}$$

This aligns with the concrete universal property of free groups discussed in example 2.2. More details on the universal property of free $\mathbf{T}$-algebras will be discussed in chapter 5.

3.2 Free Algebras for Monads

We now develop the second canonical way given by Heinrich Kleisli [Kle65] in 1965 to decompose a monad into an adjunction: the Kleisli construction.

Definition 3.6 (Kleisli Category)**.** Let $(\mathbf{T}, \eta, \mu)$ be a monad on a category $\mathcal{A}$. The *Kleisli category* $\mathcal{A}_{\mathbf{T}}$ is the category where:

1. the objects of $\mathcal{A}_{\mathbf{T}}$ are those of $\mathcal{A}$;
2. a morphism $f^\flat : X \rightsquigarrow Y$ in $\mathcal{A}_{\mathbf{T}}$ is a morphism $f : X \to \mathbf{T}Y$ in $\mathcal{A}$;
3. the composite of two morphisms $f^\flat : X \rightsquigarrow Y$, $g^\flat : Y \rightsquigarrow Z$ in $\mathcal{A}_{\mathbf{T}}$ is given in $\mathcal{A}$ by the composite $(\mu_Z \circ \mathbf{T}g \circ f)^\flat$.

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[rrr, bend right, "g^\flat \circ f^\flat"'] & \mathbf{T}Y \arrow[r, "\mathbf{T}g"] & \mathbf{T}\mathbf{T}Z \arrow[r, "\mu_Z"] & \mathbf{T}Z.
\end{tikzcd}$$

4. the identity $\mathbf{id}_X : X \rightsquigarrow X$ on an object X of $\mathcal{A}_{\mathbf{T}}$ is just $\eta_X : X \to \mathbf{T}X$ in $\mathcal{A}$.

The definition of morphisms $f^\flat$ amounts to a bijection

$$\mathcal{A}_{\mathbf{T}}(X, Y) \cong \mathcal{A}(X, \mathbf{T}Y)$$

on hom-sets. We use $\rightsquigarrow$ to denote morphisms in the Kleisli category $\mathcal{A}_{\mathbf{T}}$ and $\to$ to denote morphisms in the underlying category $\mathcal{A}$.

Theorem 3.3 (Kleisli Adjunction)**.** *Let $(\mathbf{T}, \eta, \mu)$ be a monad on a category $\mathcal{A}$. There's an adjunction $\mathbf{F}_{\mathbf{T}} \dashv \mathbf{G}_{\mathbf{T}} : \mathcal{A} \to \mathcal{A}_{\mathbf{T}}$ given by assignments* (3.14) *and* (3.15) *respectively, such that $(\mathbf{T}, \eta, \mu) = (\mathbf{G}_{\mathbf{T}} \circ \mathbf{F}_{\mathbf{T}}, \eta_{\mathbf{T}}, \mathbf{G}_{\mathbf{T}}\varepsilon_{\mathbf{T}}\mathbf{F}_{\mathbf{T}})$, where $\eta_{\mathbf{T}}$ and $\varepsilon_{\mathbf{T}}$ are the unit and counit of the adjunction, respectively.*

$$\begin{tikzcd}
X \arrow[d, "f"'] & & X \arrow[d, squiggly, "(\eta_X \circ f)^\flat"] \\
Y \arrow[r, phantom, "\overset{\mathbf{F}_{\mathbf{T}}}{\longmapsto}"] & & Y
\end{tikzcd} \tag{3.14}$$

$$\begin{tikzcd}
X \arrow[d, squiggly, "f^\flat"'] & & \mathbf{T}X \arrow[d, "\mu_Y \circ \mathbf{T}f"] \\
Y \arrow[r, phantom, "\overset{\mathbf{G}_{\mathbf{T}}}{\longmapsto}"] & & \mathbf{T}Y
\end{tikzcd} \tag{3.15}$$

Proof. A suitable large diagram shows the new composition associative. Other diagrams prove that $(\eta_X)^\flat : X \to X$ is a left and right unit for this composition. Another calculation shows that $\mathbf{F}_{\mathbf{T}}$ and $\mathbf{G}_{\mathbf{T}}$ as described are indeed functors. By construction, $f^\flat \mapsto f$ is a bijection

$$\mathcal{A}_{\mathbf{T}}(\mathbf{F}_{\mathbf{T}}X, Y) = \mathcal{A}_{\mathbf{T}}(X, Y) \cong \mathcal{A}(X, \mathbf{T}Y) = \mathcal{A}(X, \mathbf{G}_{\mathbf{T}}Y);$$

it is natural in X and Y, so yields the desired adjunction $\varphi_{\mathbf{T}}$. Its unit is η, and its counit $\varepsilon_{\mathbf{T}}$ is given by $(\varepsilon_{\mathbf{T}})_Y = (\mathbf{id}_{\mathbf{T}Y})^\flat : \mathbf{T}Y \to Y$. The resulting multiplication in $\mathcal{A}$ is $\mathbf{G_T}\varepsilon_{\mathbf{T}}\mathbf{F_T}$, which by the definition of $\mathbf{G_T}$ is exactly the given multiplication μ. Therefore the adjunction does define the original monad $\mathbf{T}$. □

For readers with familiarity with Moggi's work [Mog88] on computational λ-calculus, the Kleisli category construction can be interpreted as a way to model computations with side effects, and the morphisms in the Kleisli category represent computations that produce values along with side effects encapsulated by the monad $\mathbf{T}$. However, attentive readers might ask, how is this construction related to the title of this section, *Free* Algebras for Monads? The answer is that the Kleisli category $\mathcal{A}_{\mathbf{T}}$ is equivalent to the full subcategory generated by *free algebras* for monad $\mathbf{T}$. In this sense, there are more connections between Kleisli category and Eilenberg-Moore category beyond both being resolutions of the same monad. To see this, we first need to define what a free algebra for a monad is.

Theorem 3.4. *Let $(\mathbf{T}, \eta, \mu)$ be a monad on a category $\mathcal{A}$. Then the Kleisli category $\mathcal{A}_{\mathbf{T}}$ is equivalent to the full subcategory of $\mathcal{A}^{\mathbf{T}}$ generated by free $\mathbf{T}$-algebras.*

Proof. Denote $\mathcal{F}_{\mathbf{T}}$ for the full subcategory of $\mathcal{A}^{\mathbf{T}}$ generated by free $\mathbf{T}$-algebras, we get a functor $\mathbf{\Phi} : \mathcal{A}_{\mathbf{T}} \to \mathcal{F}_{\mathbf{T}}$ by sending every object X in $\mathcal{A}_{\mathbf{T}}$ to the free $\mathbf{T}$-algebra $(\mathbf{T}X, \mu_X)$ in $\mathcal{F}_{\mathbf{T}}$, and every morphism

$$f^\flat : X \rightsquigarrow Y$$

in $\mathcal{A}_{\mathbf{T}}$ to the $\mathbf{T}$-algebra homomorphism

$$\mathbf{\Phi}(f^\flat) = \mu_Y \circ \mathbf{T}f : \mathbf{T}X \to \mathbf{T}Y$$

$\mathbf{\Phi}(f^\flat)$ is indeed a $\mathbf{T}$-algebra homomorphism because

$$\begin{aligned}\mu_Y \circ \mathbf{T}(\mathbf{\Phi}(f^\flat)) &= \mu_Y \circ \mathbf{T}\mu_Y \circ \mathbf{TT}f \\ &= \mu_Y \circ \mu_{\mathbf{T}Y} \circ \mathbf{TT}f \\ &= \mu_Y \circ \mathbf{T}f \circ \mu_{\mathbf{T}X} \\ &= \mathbf{\Phi}f^\flat \circ \mu_X.\end{aligned}$$

Meanwhile $\mathbf{\Phi}$ is indeed a functor because it preserves identity

$$\mathbf{\Phi}(\eta_X) = \mu_X \circ \mathbf{T}\eta_X = \mathbf{id}_{\mathbf{T}X},$$

and composition

$$\begin{aligned}\mathbf{\Phi}(g^\flat \circ f^\flat) &= \mu_Z \circ \mathbf{T}g \circ \mu_Y \circ \mathbf{T}f \\ &= \mu_Z \circ \mu_{\mathbf{T}Z} \circ \mathbf{TT}g \circ \mathbf{T}f \\ &= \mu_Z \circ \mathbf{T}\mu_Z \circ \mathbf{TT}g \circ \mathbf{T}f \\ &= \mu_Z \circ \mathbf{T}(\mu_Z \circ \mathbf{T}g \circ f) \\ &= \mathbf{\Phi}(g^\flat) \circ \mathbf{\Phi}(f^\flat).\end{aligned}$$

By the choice of $\mathcal{F}_\mathbf{T}$, every object of $\mathcal{F}_\mathbf{T}$ is isomorphic to an object of the form $\mathbf{\Phi}X$, thus it remains to prove that $\mathbf{\Phi}$ is full and faithful. If $f, g : X \rightrightarrows \mathbf{T}Y$ are such that

$$\mu_Y \circ \mathbf{T}f = \mu_Y \circ \mathbf{T}g$$

then

$$f = \mu_Y \circ \eta_{\mathbf{T}Y} \circ f = \mu_Y \circ \mathbf{T}f \circ \eta_X = \mu_Y \circ \mathbf{T}g \circ \eta_X = \mu_Y \circ \eta_{\mathbf{T}Y} \circ g = g,$$

so that $\mathbf{\Phi}$ is faithful. Φ is also full since, given

$$h : (\mathbf{T}X, \mu_X) \to (\mathbf{T}Y, \mu_Y)$$

in $\mathcal{F}_\mathbf{T}$, the composite $h \circ \eta_X$ is such that

$$\mu_Y \circ \mathbf{T}(h \circ \eta_X) = \mu_Y \circ \mathbf{T}h \circ \mathbf{T}\eta_X = h \circ \mu_X \circ \mathbf{T}\eta_X = h.$$

□

3.3 Resolving Monads into Adjunctions

The two constructions given by Kleisli and Eilenberg-Moore not only provide us with two different ways to resolve a monad into an adjunction, they are respectively *initial* and *terminal* in a certain sense. In this section we will see, for any monad $(\mathbf{T}, \eta, \mu)$ on a category $\mathcal{A}$, there is a category describing all such resolutions of the monad $\mathbf{T}$, called the *category of resolutions*, and the Kleisli and Eilenberg-Moore constructions are respectively initial and terminal objects in this category.

Definition 3.7 (Category of Resolutions)**.** The *category of resolutions* $\mathbf{Adj_T}$ of a monad $(\mathbf{T}, \eta, \mu)$ on a category $\mathcal{A}$ is the category whose objects are fully-specified adjunctions $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ with

$$\begin{tikzcd} \mathcal{A} \arrow[r, shift left, "\mathbf{F}"] \arrow[r, phantom, "\perp"] & \mathcal{B} \arrow[l, shift left, "\mathbf{G}"] \end{tikzcd} \qquad \eta : \mathbf{id}_\mathcal{A} \to \mathbf{GF}, \qquad \varepsilon : \mathbf{FG} \to \mathbf{id}_\mathcal{B}$$

inducing the monad $\mathbf{T}$ on $\mathcal{A}$. A morphism from $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ to $\mathbf{F}' \dashv \mathbf{G}' : \mathcal{A} \to \mathcal{B}'$ is a functor $\mathbf{H} : \mathcal{B} \to \mathcal{B}'$ making the diagram

$$\begin{tikzcd}
\mathcal{B} \arrow[rr, "\mathbf{H}"] \arrow[dr, shift left, "\mathbf{G}"] & & \mathcal{B}' \arrow[dl, shift left, "\mathbf{G}'"] \\
& \mathcal{A} \arrow[ul, shift left, "\mathbf{F}"] \arrow[ul, phantom, "\top"] \arrow[ur, shift left, "\mathbf{F}'"] \arrow[ur, phantom, "\perp"] &
\end{tikzcd}$$

commute with both left and right adjoints. That is $\mathbf{H} \circ \mathbf{F} = \mathbf{F}'$ and $\mathbf{G} \circ \mathbf{H} = \mathbf{G}'$. $\mathbf{H}$ is called the *comparison functor* from adjunction $\mathbf{F} \dashv \mathbf{G}$ to $\mathbf{F}' \dashv \mathbf{G}'$.

We say that Kleisli and Eilenberg-Moore constructions are respectively initial and terminal in $\mathbf{Adj_T}$ is because comparison functors from Kleisli category to any other resolution exist uniquely, and so are comparison functors from any other resolution to Eilenberg-Moore category, as stated in the following theorem.

Theorem 3.5 (Comparison Functor)**.** *Let* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ *be an adjunction with associated monad* $(\mathbf{T}, \eta, \mu)$ *on* $\mathcal{A}$ *(so* $\mathbf{T} = \mathbf{GF}$, $\mu = \mathbf{G}\varepsilon\mathbf{F}$*). Then there exist* unique *functors*

$$\mathbf{K} : \mathcal{B} \to \mathcal{A}^{\mathbf{T}} \qquad \textit{and} \qquad \mathbf{J} : \mathcal{A}_{\mathbf{T}} \to \mathcal{B}$$

such that diagram (3.16) *commutes:*

$$\begin{tikzcd}
\mathcal{A}_{\mathbf{T}} \arrow[r, dashed, "\mathbf{J}", "\exists!"'] \arrow[dr, shift left, "\mathbf{G_T}"] & \mathcal{B} \arrow[r, dashed, "\mathbf{K}", "\exists!"'] \arrow[d, shift left, "\mathbf{G}"] & \mathcal{A}^{\mathbf{T}} \arrow[dl, shift left, "\mathbf{G^T}"] \\
& \mathcal{A} \arrow[ul, shift left, "\mathbf{F_T}"] \arrow[ul, phantom, "\top"] \arrow[u, shift left, "\mathbf{F}"] \arrow[u, phantom, "\dashv"] \arrow[ur, shift left, "\mathbf{F^T}"] \arrow[ur, phantom, "\perp"] &
\end{tikzcd} \tag{3.16}$$

Moreover, $\mathbf{F_T} \dashv \mathbf{G_T}$ *is initial in* $\mathbf{Adj_T}$ *and* $\mathbf{F^T} \dashv \mathbf{G^T}$ *is terminal in* $\mathbf{Adj_T}$*.*

Proof. We first prove the existence of the comparison functors, then show they are unique. Consider functor $\mathbf{K} : \mathcal{B} \to \mathcal{A}^{\mathbf{T}}$ defined by:

$$\mathbf{K}B \triangleq (\mathbf{G}B, \mathbf{G}\varepsilon_B : \mathbf{GFG}B \to \mathbf{G}B) \qquad \mathbf{K}(A \xrightarrow{f} B) \triangleq \mathbf{G}A \xrightarrow{\mathbf{G}f} \mathbf{G}B.$$

Claim 3.1. *$\mathbf{K}$ is a well-defined functor $\mathcal{B} \to \mathcal{A}^{\mathbf{T}}$ making diagram (3.16) commute.*

(1) *Unit property is satisfied, i.e. diagram (3.8) commutes for $\mathbf{T} = \mathbf{GF}$*

$$\alpha_B \circ \eta_{\mathbf{G}B} = \mathbf{G}\varepsilon_B \circ \eta_{\mathbf{G}B} = \mathbf{id}_{\mathbf{G}B}$$

by the triangular identity $\mathbf{G}\varepsilon \circ \eta\mathbf{G} = \mathbf{id}_{\mathbf{G}}$.

(2) *Action property is satisfied, i.e. diagram (3.9) commutes for $\mathbf{T} = \mathbf{GF}$*

$$\alpha_B \circ \mu_{\mathbf{G}B} = \mathbf{G}\varepsilon_B \circ \mathbf{G}\varepsilon_{\mathbf{FG}B} = \mathbf{G}\left(\varepsilon_B \circ \varepsilon_{\mathbf{FG}B}\right) = \mathbf{G}\left(\varepsilon_B \circ \mathbf{FG}\varepsilon_B\right) = \mathbf{G}\varepsilon_B \circ \mathbf{T}\alpha_B,$$

where the middle equality is the horizontal composition for ε in an adjunction, i.e. the monad associativity $\mu \circ \mathbf{T}\mu = \mu \circ \mu\mathbf{T}$ unfolded at $\mathbf{G}B$.

$\mathbf{K}$ *also preserves algebra morphisms. For $f : B \to B'$,*

$$\mathbf{K}f \circ \alpha_B = \mathbf{G}f \circ \mathbf{G}\varepsilon_B = \mathbf{G}\varepsilon_{B'} \circ \mathbf{GFG}f = \alpha_{B'} \circ \mathbf{TG}f,$$

by naturality of $\varepsilon : \mathbf{FG} \Rightarrow \mathbf{id}_{\mathcal{B}}$.

To see the uniqueness of $\mathbf{K}$, suppose $\mathbf{K}' : \mathcal{B} \to \mathcal{A}^{\mathbf{T}}$ satisfies $\mathbf{G}^{\mathbf{T}}\mathbf{K}' = \mathbf{G}$ and $\mathbf{K}'\mathbf{F} = \mathbf{F}^{\mathbf{T}}$. Then for each $a \in \mathcal{B}$, the underlying object of $\mathbf{K}'a$ must be $\mathbf{G}a$, so $\mathbf{K}'a = (\mathbf{G}a, h_a)$ for some $h_a : \mathbf{TG}a \to \mathbf{G}a$. Because $\mathbf{K}'$ is a *map of adjunctions* from $(\mathbf{F}, \mathbf{G}, \eta, \varepsilon)$ to $(\mathbf{F}^{\mathbf{T}}, \mathbf{G}^{\mathbf{T}}, \eta, e^{\mathbf{T}})$ with identity on $\mathcal{A}$, we have the compatibility

$$\mathbf{K}'\varepsilon = e^{\mathbf{T}}\mathbf{K}',$$

hence at a we get $h_a = \mathbf{G}\varepsilon_a$. Thus $\mathbf{K}' = \mathbf{K}$ objectwise and on morphisms ($\mathbf{G}^{\mathbf{T}}\mathbf{K}' = \mathbf{G}$ forces $\mathbf{K}'f = \mathbf{G}f$). Therefore $\mathbf{K}$ is unique.

Similarly, we construct $\mathbf{J} : \mathcal{A}_{\mathbf{T}} \to \mathcal{B}$ as follows.

$$\mathbf{J}A \triangleq \mathbf{F}A \qquad \mathbf{J}(A \xrightarrow{f} \mathbf{T}B) \triangleq \mathbf{F}A \xrightarrow{\varepsilon_{\mathbf{F}B} \circ \mathbf{F}f} \mathbf{F}B.$$

the proof that $\mathbf{J}$ is a well-defined functor making diagram (3.16) commute and that $\mathbf{J}$ is unique is a routine check similar to that of $\mathbf{K}$. □

Combining theorem 3.4 and the definition of comparison functor $\mathbf{K} : \mathcal{B} \to \mathcal{A}^{\mathbf{T}}$, we immediately get the following proposition.

Proposition 3.1. *Let $\mathcal{B} = \mathcal{A}_{\mathbf{T}}$, then the comparison functor $\mathbf{K} : \mathcal{B} \to \mathcal{A}^{\mathbf{T}}$ is full and faithful and its image consists of free $\mathbf{T}$-algebras.*

We conclude this chapter with the definition of *monadicity*, the main topic of this thesis, which describes a property that metrizes how much an adjunction behaves like the canonical adjunction

$$\mathbf{F}^{\mathbf{T}} \dashv \mathbf{G}^{\mathbf{T}} : \mathcal{A} \to \mathcal{A}^{\mathbf{T}}$$

induced by its associated monad $\mathbf{T}$.

Definition 3.8 (Monadicity [Rie17])**.** An adjunction $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ is called *monadic* if the comparison functor $\mathbf{K} : \mathcal{B} \to \mathcal{A}^{\mathbf{T}}$ is an equivalence of categories. Moreover, a functor $\mathbf{G} : \mathcal{B} \to \mathcal{A}$ is called *monadic* if it admits a left adjoint $\mathbf{F}$ such that $\mathbf{F} \dashv \mathbf{G}$ is monadic. If the equivalence is further an isomorphism, then the adjunction is called *strictly monadic*, and so is the functor $\mathbf{G}$.

Chapter 4

Monadicity Theorems

We gave the definition of monadic adjunctions and monadic functors in the end of chapter 3, which are adjunctions that behave like the free-forgetful adjunctions arising from the Eilenberg-Moore construction. However, several natural questions remain:

(1) Why do we care about this?

(2) How can we tell if a given adjunction is monadic?

To answer the first question, an intuitive answer is that monadic adjunctions reveal deep connections between seemingly different mathematical structures. In fact, every category determined by an algebraic theory presented by operations and equations, for example **Grp** of groups and **Mon** of monoids, has a free and forgetful adjunction over **Set** that is monadic. Moreover, beyond these algebraic examples, some nonalgebraic categories, such as **CHaus** of compact Hausdorff spaces, are also monadic. The connection between monadicity and algebraicity will be elaborated in chapter 5.

As for the second question, Jonathan Mock Beck provided a criterion in his Ph.D. thesis [Bec03] in 1967, originally known as *Tripleability Theorem* (because monads were called triples at that time), and later popularized by Michael Barr and Charles Wells in their book *Toposes, Triples and Theories* [BW00]. It turns out that whether an adjunction is monadic can be determined by certain preservation and exactness conditions, now known as the *Barr-Beck Monadicity Theorem*. In this chapter, we will provide detailed intuition on *coequalizers* in section 4.1, which is essential to understanding the exactness conditions in the Barr-Beck Monadicity Theorem, and then present the theorem itself with various weaker variants in section 4.2. Finally, we illustrate the utility of these monadicity theorems by presenting several examples of monadic adjunctions in section 4.3.

4.1 Epimorphisms and Coequalizers

We begin with epimorphisms and coequalizers, since they play an important role in a family of monadicity theorems. A helpful analogy is the First Isomorphism Theorem in group theory: a surjective group homomorphism $\varphi\colon G \to H$ induces an isomorphism

$$G/\ker\varphi \cong \operatorname{Im}\varphi = H.$$

We know that essentially the same statement holds for rings and modules.

As we have seen in sections 2.1 and 2.2, epimorphisms capture and generalize the essence of surjective maps, while coequalizers capture and generalize the essence of quotient constructions. Beck's Monadicity Theorem says that a left adjoint **G** is monadic if and only if it respects such quotient structure to an appropriate extent. More precisely, different monadicity theorems focus on different kinds of quotient structures, and these correspond to different kinds of coequalizers. In this section, we study and clarify the relationships among these various notions of epimorphism and coequalizer, aimed at providing an intuitive understanding of their roles in the monadicity theorems presented later in this chapter.

4.1.1 Coequalizers

The concept of coequalizer in a general category is the generalization of the construction where for two functions $f, g : X \to Y$ between sets X and Y, one forms the set $Y/\sim$ of equivalence classes induced by the equivalence relation generated by the relation $f(x) \sim g(x)$ for all $x \in X$. This means that the quotient function $q : Y \to Y/\sim$ satisfies

$$q \circ f = q \circ g$$

In this form this may be phrased generally in any category.

Recall in definition 2.14 we have defined a coequalizer of a parallel pair of morphisms in terms of a colimit and universal property, but in order to understand the role of coequalizers in monadicity, we need to understand the intrinsic properties of coequalizers in more detail. Coequalizers are defined on a parallel pair of morphisms called a *fork*.

Definition 4.1 (fork)**.** A *fork* in a category $\mathcal{A}$ is a diagram

$$\begin{tikzcd} X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y \arrow[r, "q"] & Q \end{tikzcd} \tag{4.1}$$

in $\mathcal{A}$ with $q \circ f = q \circ g$, where $f, g : X \rightrightarrows Y$ are morphisms in $\mathcal{A}$. Sometimes we call diagram (4.1) a *cofork* in order to distinguish from the diagram (4.2)

$$\begin{tikzcd} A \arrow[r, "a"] & X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y \end{tikzcd} \tag{4.2}$$

Through this thesis, we use the term *fork* to refer to diagram (4.1).

When f and g are clear from the context, we also call the morphism q a *fork* of f and g. Obviously there is an underlying fork in the diagram of a coequalizer. We can make this more explicit as follows.

Definition 4.2 (Coequalizer)**.** Let

$$\begin{tikzcd} X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y \arrow[r, "q"] & Q \end{tikzcd} \tag{4.3}$$

be a fork in a category $\mathcal{A}$. This diagram is said to be a *coequalizer* of f and g if q is universal for this property, that is, for any other morphism $h : Y \to Z$ such that $h \circ f = h \circ g$, there exists a unique morphism $z : Q \to Z$ such that $h = z \circ q$. In particular, diagram (4.4) commutes.

$$\begin{tikzcd} X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y \arrow[r, "q"] \arrow[dr, "h"'] & Q \arrow[d, dashed, "z"] \\ & & Z \end{tikzcd} \tag{4.4}$$

Similarly, when f and g are clear from the context, we also call q the *coequalizer* of f and g, denoted by $q = \mathrm{coeq}(f, g)$. Coequalizers are colimits, thus unique up to isomorphism when they exist.

Example 4.1 (Coequalizers in **Grp**)**.** Let us consider the category of groups **Grp**. Given a parallel pair of group homomorphisms $f, g : X \rightrightarrows Y$, the coequalizer is given by quotient map $q = y \mapsto yN$ from Y to the quotient group Y/N where N is the normal closure in group Y defined as:

$$N = \mathrm{ncl}_Y\{f^{-1}(x)g(x) \mid x \in X\}$$

We need to check the two requirements of q to be a coequalizer:

(1) $q \circ f = q \circ g$: For any $x \in X$, we have $q(f(x)) = f(x)N$ and $q(g(x)) = g(x)N$. Since $f^{-1}(x)g(x) \in N$, we have $f(x)N = g(x)N$.

(2) Universal property: Given any group homomorphism $h : Y \to Z$ such that $h \circ f = h \circ g$, we need to show that there exists a unique homomorphism $\overline{h} : Y/N \to Z$ such that $h = \overline{h} \circ q$. This is true because $N \subseteq \ker h$, so h factors uniquely through the quotient map.

4.1.2 Epimorphisms

A coequalizer can be intuitively understood as the quotient or projection map, induced by identifying elements according to the enforced relation. Since quotient maps are known to be surjective in **Set** or **Grp**, and this is also the case in general categories. In particular, coequalizers are epimorphisms.

Definition 4.3 (Regular Epimorphism)**.** A morphism $h : Y \to Z$ is called a *regular epimorphism* if it is the coequalizer of some parallel pair of morphisms $f, g : X \rightrightarrows Y$.

$$\begin{tikzcd} X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y \arrow[r, "h"] & Z \end{tikzcd} \tag{4.5}$$

Lemma 4.1. *Regular epimorphisms are epimorphisms.*

Proof. Let $h : Y \to Z$ be the coequalizer of $f, g : X \rightrightarrows Y$. To show h is an epimorphism, we must show that for any pair of morphisms $k_1, k_2 : Z \to W$, if $k_1 \circ h = k_2 \circ h$, then $k_1 = k_2$. The equation $k_1 \circ h = k_2 \circ h$ implies that the morphism $k_1 \circ h : Y \to W$ coequalizes f and g, since $(k_1 \circ h) \circ f = k_1 \circ (h \circ f) = k_1 \circ (h \circ g) = (k_1 \circ h) \circ g$. By the universal property of the coequalizer h, there exists a *unique* morphism $z : Z \to W$ such that $z \circ h = k_1 \circ h$. Both k_1 and k_2 are candidates for this unique morphism z. Therefore, by uniqueness, we must have $k_1 = k_2$. □

Regular epimorphisms are indeed epimorphisms, and an even stronger class of epimorphisms is *split epimorphisms*, which have a section (i.e. a right inverse).

Definition 4.4 (Split Epimorphism)**.** A *split epimorphism* in a category $\mathcal{A}$ is a morphism $e : A \to B$ in $\mathcal{A}$ such that e has a section $s : B \to A$. In this case, we say e is a *retraction* of s, B is a *retract* of A, and the pair (e, s) is a splitting of the idempotent $s \circ e : A \to A$.

Lemma 4.2. *Split epimorphisms are regular epimorphisms.*

Proof. Every split epimorphism $e : A \to B$ with section $s : B \to A$ automatically comes with the coequalizer of the parallel pair $(\mathrm{id}_A, s \circ e)$:

$$\begin{tikzcd}
A \arrow[r, shift left, "s\circ e"] \arrow[r, shift right, "\mathrm{id}_A"'] & A \arrow[r, "e"] & B \arrow[l, bend left, "s"]
\end{tikzcd} \tag{4.6}$$

□

Fix a category $\mathcal{A}$, let E, R and S be the class of epimorphisms, regular epimorphisms and split epimorphisms in $\mathcal{A}$ respectively. We have the following hierarchy:

$$\begin{tikzcd}
E \arrow[d, dash, "\rotatebox{90}{$\subseteq$}"] \\
R \arrow[d, dash, "\rotatebox{90}{$\subseteq$}"] \\
S
\end{tikzcd}$$

4.1.3 Absolute, Split and Reflexive Coequalizers

A natural next question is how these structures behave when a functor is applied, and in this section we care about functorial actions over coequalizers in particular. A better scenario for a coequalizer is being stable under the action of more functors, and the best such coequalizers are called *absolute* coequalizers. There are three main actions of functorials that we consider: preservation, reflection, and creation.

Definition 4.5 (Actions of Functors over Coequalizers)**.** Fix a pair of morphisms $f, g : X \rightrightarrows Y$ in $\mathcal{A}$. A functor $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ is said to

(1) *preserve* coequalizers for this pair if

$$\begin{gathered} q \text{ is a coequalizer of } f, g \\ \Downarrow \\ \mathbf{F}q \text{ is a coequalizer of } \mathbf{F}f, \mathbf{F}g \end{gathered}$$

(2) *detect* coequalizers for this pair if

$$\begin{gathered} \mathbf{F}f, \mathbf{F}g \text{ has a coequalizer} \\ \Downarrow \\ f, g \text{ has a coequalizer} \end{gathered}$$

(3) *reflect* coequalizers for this pair if

$$\begin{gathered} q \text{ coequalizes } f, g \text{ and } \mathbf{F}q \text{ is a coequalizer of } \mathbf{F}f, \mathbf{F}g \\ \Downarrow \\ q \text{ is a coequalizer of } f, g \end{gathered}$$

(4) *create* coequalizers for this pair if

$$\begin{gathered} \mathbf{F}f, \mathbf{F}g \text{ has a coequalizer } q \\ \Downarrow \\ \exists! \text{ fork } p \text{ of } f, g \text{ s.t. } p \text{ is a coequalizer of } f, g \text{ and } \mathbf{F}p \cong q \end{gathered}$$

where $\exists!$ means unique existence up to isomorphism. Moreover if the existence is unique up to equality, then we say $\mathbf{F}$ *strictly creates* coequalizers for this pair.

Note the subtlety between *creation* and *strict creation* in definition 4.5: *Strict creation* requires the lifted coequalizer to be exactly the given one under the functor, while *creation* only requires them to be isomorphic.

The definition of *preservation* and *reflection* of coequalizers are intuitive, but *creation* of coequalizers is a bit more involved. In fact, there is an equivalent way to define creation of coequalizers that is much more clear and reveals the relationship among these actions more clearly.

Lemma 4.3. *Let S be a collection of parallel pairs of morphisms in a category $\mathcal{A}$, and let $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ be a functor. Then (1) and (2) are equivalent.*

(1) *$\mathcal{A}$ has, $\mathbf{F}$ preserves and reflects coequalizers for all pairs in S.*

(2) *$\mathcal{B}$ has coequalizers for all pairs $(\mathbf{F}f, \mathbf{F}g)$ with $(f, g) \in S$, and $\mathbf{F}$ creates coequalizers for all pairs in S.*

Proof. Fix a pair $(f, g) \in S$ with $f, g : X \rightrightarrows Y$.

(1) ⇒ (2). Assume (1). Since $\mathcal{A}$ has coequalizers for pairs in S, pick a coequalizer $p : Y \to A$ of f, g in $\mathcal{A}$. Because $\mathbf{F}$ preserves these coequalizers, the morphism

$$\mathbf{F}p : \mathbf{F}Y \longrightarrow \mathbf{F}A$$

is a coequalizer of $\mathbf{F}f, \mathbf{F}g$ in $\mathcal{B}$, so $\mathcal{B}$ has coequalizers of all pairs $(\mathbf{F}f, \mathbf{F}g)$ with $(f, g) \in S$.

To see the creation, suppose $q : \mathbf{F}Y \to Q$ is a coequalizer of $\mathbf{F}f, \mathbf{F}g$ in $\mathcal{B}$, then q and $\mathbf{F}p$ are isomorphic via an isomorphism $\alpha : Q \to \mathbf{F}A$, and p is such a coequalizer of f, g in $\mathcal{A}$ that $\mathbf{F}p \cong q$. with $\alpha \circ q = \mathbf{F}p$. Equivalently, $\alpha^{-1} : \mathbf{F}A \to Q$ is an isomorphism such that $\alpha^{-1} \circ \mathbf{F}p = q$. To see the uniqueness of p as a fork of f, g, Let $h : Y \to C$ be any morphism in $\mathcal{A}$ such that $\mathbf{F}h$ is a coequalizer of $\mathbf{F}f, \mathbf{F}g$ in $\mathcal{B}$. Since $\mathbf{F}$ reflects coequalizers by (1), it follows that h is a coequalizer of f, g in $\mathcal{A}$. Hence (2) holds.

(2) ⇒ (1). Assume (2). For our fixed pair $(f, g) \in S$, choose a coequalizer $q : \mathbf{F}Y \to Q$ of $\mathbf{F}f, \mathbf{F}g$ in $\mathcal{B}$ which exists by assumption. By the creation property, there is a coequalizer $p : Y \to A$ of f, g in $\mathcal{A}$ and an isomorphism $\alpha : \mathbf{F}A \to Q$ with $\alpha \circ \mathbf{F}p = q$. Thus $\mathcal{A}$ has a coequalizer of f, g.

To see that $\mathbf{F}$ preserves these coequalizers, let $p' : Y \to A'$ be any coequalizer of f, g in $\mathcal{A}$. Applying $\mathbf{F}$ we obtain $\mathbf{F}p' : \mathbf{F}Y \to \mathbf{F}A'$. Since coequalizers are unique up to unique isomorphism in $\mathcal{B}$ and q is a coequalizer of $\mathbf{F}f, \mathbf{F}g$, there exists an isomorphism $\beta : \mathbf{F}A \to \mathbf{F}A'$ such that $\beta \circ \mathbf{F}p = \mathbf{F}p'$. Because $\mathbf{F}p$ is a coequalizer of $\mathbf{F}f, \mathbf{F}g$, so is $\mathbf{F}p'$. Hence $\mathbf{F}$ preserves the coequalizer of f, g.

Finally, to see that $\mathbf{F}$ reflects coequalizers for (f, g), let $h : Y \to C$ be a fork such that $\mathbf{F}h$ is a coequalizer of $\mathbf{F}f, \mathbf{F}g$ in $\mathcal{B}$. By creation, h is then a coequalizer of f, g in $\mathcal{A}$. Since (f, g) was arbitrary in S, $\mathcal{A}$ has coequalizers for all pairs in S, and $\mathbf{F}$ preserves and reflects them. □

Lemma 4.3 can be intuitively understood as follows: assuming that coequalizers exist (for the relevant pairs) in both the domain and codomain categories, a functor preserves and reflects those coequalizers if and only if it creates them.

What are coequalizers that behave well under functorial actions? One of the robust coequalizers are those that are preserved by *all* functors. This property is called *absoluteness*.

Definition 4.6 (Absolute Coequalizer)**.** A coequalizer is called *absolute* if it is preserved by all functors.

Definition 4.6 itself, however, does not provide us any information about which coequalizers are absolute. However, by adding a few conditions to coequalizers or even forks, will give rise to some well-behaved absolute coequalizers. A *split fork* or *split coequalizer* is a sufficient condition for being an absolute coequalizer.

Definition 4.7 (Split Coequalizer)**.** A *split fork* is a fork (4.1) with two additional *splitting* morphisms

$$\begin{tikzcd} X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y \arrow[r, "q"] \arrow[l, bend left, "s"] & Q \arrow[l, bend left, "t"] \end{tikzcd} \tag{4.7}$$

such that

$$q \circ f = q \circ g \qquad f \circ s = \mathrm{id}_B \qquad g \circ s = t \circ q \qquad q \circ t = \mathrm{id}_Q$$

A split fork is also called a *split coequalizer*. When it is clear from the context, we also call q the split coequalizer of this split fork. Note that a split coequalizer is indeed a coequalizer.

Proof of well-definedness. Let the split fork be given in diagram (4.7), with splitting morphisms $s : Y \to X$ and $t : Q \to Y$. Suppose we have a morphism $h : Y \to Z$ such that $h \circ f = h \circ g$.

Claim 4.1. *$z = h \circ t : Q \to Z$ is the unique morphism such that $h = z \circ q$.*

$$\begin{aligned} z \circ q &= (h \circ t) \circ q \\ &= h \circ (t \circ q) \\ &= h \circ (g \circ s) \\ &= (h \circ g) \circ s \\ &= (h \circ f) \circ s \\ &= h \circ (f \circ s) \\ &= h \circ \mathrm{id}_Y \\ &= h \end{aligned}$$

To see the uniqueness, suppose there is another morphism $z' : Q \to Z$ such that $h = z' \circ q$. Then:

$$\begin{aligned} z &= h \circ t \\ &= (z' \circ q) \circ t \\ &= z' \circ (q \circ t) \\ &= z' \circ \mathrm{id}_Q \\ &= z' \end{aligned}$$

□

Lemma 4.4. *Split coequalizers are absolute coequalizers.*

Proof. Let $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ be any functor and let $q : Y \to Q$ be a split coequalizer of $f, g : X \to Y$ with splitting morphisms $s : Y \to X$ and $t : Q \to Y$. The defining identities are

$$q \circ t = \mathbf{id}_Q \qquad f \circ s = \mathbf{id}_Y \qquad g \circ s = t \circ q$$

Applying the functor $\mathbf{F}$ preserves these identities:

$$\mathbf{F}q \circ \mathbf{F}t = \mathbf{id}_{\mathbf{F}Q} \qquad \mathbf{F}f \circ \mathbf{F}s = \mathbf{id}_{\mathbf{F}Y} \qquad \mathbf{F}g \circ \mathbf{F}s = \mathbf{F}t \circ \mathbf{F}q$$

This means the resulting diagram in $\mathcal{B}$ is also a split fork, and thus a coequalizer. In particular, $\mathbf{F}q$ is a split coequalizer of $\mathbf{F}f$ and $\mathbf{F}g$ in $\mathcal{B}$. □

The notion of *splitting* is ubiquitous in algebra: it captures when a morphism admits a *right inverse*. Readers with familiarity with homological algebra may have seen this in the context of *split short exact sequences*. In fact, a split short exact sequences give rise to a split coequalizer, and the reason is the underlying split epimorphism.

Example 4.2 (Split Epimorphisms give rise to split coequalizers)**.** Given a category $\mathcal{A}$ and a *split epimorphism* $\pi : B \to C$ with a section $s : C \to B$. Then π is a coequalizer of the parallel pair

$$s \circ \pi, \mathrm{id}_B : B \rightrightarrows B.$$

In particular, the diagram (4.8) is a split fork:

$$\begin{tikzcd}
B \arrow[r, shift left, "\mathrm{id}_B"] \arrow[r, shift right, "s\circ\pi"'] & B \arrow[r, "\pi"] \arrow[l, bend left=60, "\mathrm{id}_B"] & C \arrow[l, bend left=60, "s"]
\end{tikzcd} \tag{4.8}$$

The proof is a straightforward check of the four conditions of a split fork in definition 4.7. Let $f = \mathrm{id}_B$, $g = s \circ \pi$, $q = \pi$, $t = \mathrm{id}_B$, and s be the section of π. We verify the required identities:

(1) $q \circ f = \pi \circ \mathrm{id}_B = \pi = \pi \circ (s \circ \pi) = q \circ g$

(2) $f \circ s = \mathrm{id}_B \circ s = s$

(3) $g \circ s = (s \circ \pi) \circ s = s \circ (\pi \circ s) = s \circ \mathrm{id}_C = s = t \circ q$

(4) $q \circ t = \pi \circ \mathrm{id}_B = \pi = \mathrm{id}_C$

This construction also aligns with lemma 4.2 which states that split epimorphisms are regular epimorphisms.

In fact, split epimorphisms appear quite frequently in algebraic contexts. For example consider a split short exact sequence of R-modules for a ring R:

$$\begin{tikzcd}
1 \arrow[r] & A \arrow[r, "\varphi"] & B \arrow[r, "\psi"] & C \arrow[r] \arrow[l, bend left, "s"] & 1
\end{tikzcd}$$

with a section $s : C \to B$ of ψ, a known result in homological algebra [1] states that $\psi : B \to C$ is a epimorphism, then ψ is a split epimorphism, and a similar argument as in example 4.2 shows that ψ gives rise to a split coequalizer.

$$\begin{tikzcd}
B \arrow[r, shift left, "\mathrm{id}_B"] \arrow[r, shift right, "s\circ\psi"'] & B \arrow[r, "\psi"] \arrow[l, bend left=60, "\mathrm{id}_B"] & C \arrow[l, bend left, "s"]
\end{tikzcd}$$

One of the most important examples of split coequalizers arises from monad algebras, called *Beck's coequalizers*. Every monad algebra induces a split coequalizer diagram in the underlying category.

Example 4.3 (Beck's Coequalizer)**.** Given a monad $(\mathbf{T}, \eta, \mu)$ on a category $\mathcal{A}$, any monad algebra $(A, \alpha : \mathbf{T}A \to A)$ defines a split coequalizer diagram (4.3) in $\mathcal{A}$:

$$\begin{tikzcd}
\mathbf{TT}A \arrow[r, shift left, "\mathbf{T}\alpha"] \arrow[r, shift right, "\mu_A"'] & \mathbf{T}A \arrow[r, "\alpha"] \arrow[l, bend left=60, "\eta_{\mathbf{T}A}"] & A \arrow[l, bend left, "\eta_A"]
\end{tikzcd} \tag{4.9}$$

To see that this is indeed a split coequalizer, we verify the four required identities:

(1) $\alpha \circ \mathbf{T}\alpha = \alpha \circ \mu_A$ by the associativity axiom of monad algebras.

(2) $\mathbf{T}\alpha \circ \eta_{\mathbf{T}A} = \mathrm{id}_{\mathbf{T}A}$ by the unit axiom of monad algebras.

(3) $\mu_A \circ \eta_{\mathbf{T}A} = \mathrm{id}_{\mathbf{T}A}$ by the monad axiom.

(4) $\alpha \circ \eta_A = \mathrm{id}_A$ by the unit axiom of monad algebras.

This example also illustrates an important conclusion that aligns with what we have learnt in algebra courses: every (monad) algebra is a quotient of some free (monad) algebra.

[1] We did not define exact sequence rigorously in Category Theory

In addition to split coequalizers, another important class of well-behaved coequalizers are *reflexive coequalizers*, which are coequalizers of *reflexive pairs*.

Definition 4.8 (Reflexive Pair)**.** A *reflexive pair* is a parallel pair $f, g : A \rightrightarrows B$ having a common section, that is, a morphism $s : B \to A$ such that $f \circ s = g \circ s = \mathrm{id}_B$. The coequalizer of a reflexive pair is called a *reflexive coequalizer*.

Why do we care about these variants of coequalizers? It turns out that variants of *monadicity theorems*, which we will discuss in section 4.2, often require different functorial behaviors of different types of coequalizers. For instance, the *Reflexive Monadicity Theorem* requires the existence and preservation of reflexive coequalizers by the relevant functor.

4.2 Monadicity Theorems

In this section, we present several formulations of the *monadicity* (*tripleability*) *theorem*, which gives necessary and sufficient conditions for a functor to be monadic. We begin with Beck's classical theorem and then discuss variations.

4.2.1 Weak Monadicity Theorems

We now establish a convenient criterion, the *Weak Monadicity Theorem* or **Wtt**[2], originally due to Beck [Bec03], that breaks the verification of monadicity into parts.

Theorem 4.1 (Weak Monadicity Theorem, **Wtt** [Bec03, Theorem 1])**.** *Let* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ *be an adjunction, then* $\mathbf{G}$ *is monadic if*

(1) *$\mathcal{B}$ has all coequalizers.*

(2) *$\mathbf{G}$ preserves all coequalizers.*

(3) *$\mathbf{G}$ reflects all isomorphisms.*

A functor is called conservative *if it reflects all isomorphisms.*

Proof. The proof strategy is by showing that

(1) ⇒ the comparison functor $\mathbf{K} : \mathcal{B} \to \mathcal{A}^{\mathbf{T}}$ admits a left adjoint $\mathbf{L}$.

(1)(2) ⇒ the unit $\eta^{\mathbf{L}} : \mathbf{id}_{\mathcal{A}^{\mathbf{T}}} \Longrightarrow \mathbf{KL}$ is a natural isomorphism.

[2] The first "T" stands for *Tripleability*, an antiquated synonym for Monadicity

(1)(2)(3) $\Rightarrow$ the counit $\varepsilon^{\mathbf{L}} : \mathbf{LK} \Longrightarrow \mathbf{id}_{\mathcal{B}}$ is a natural isomorphism.

(1) We construct the left adjoint $\mathbf{L} : \mathcal{A}^{\mathbf{T}} \to \mathcal{B}$ by exhibiting a natural isomorphism for each $\mathbf{T}$-algebra (A, α) and object B in $\mathcal{B}$:

$$\mathcal{A}^{\mathbf{T}}((A, \alpha), \mathbf{K}-) \cong \mathcal{B}(\mathbf{L}(A, \alpha), -)$$

Recall from theorem 3.5 that the comparison functor $\mathbf{K} : \mathcal{B} \to \mathcal{A}^{\mathbf{T}}$ is defined by

$$\mathbf{K}B = (\mathbf{G}B, \mathbf{G}\varepsilon_B : \mathbf{TG}B \to \mathbf{G}B)\,.$$

so a morphism $h \in \mathcal{A}^{\mathbf{T}}((A, \alpha), \mathbf{K}B)$ is determined by its underlying arrow $h : A \to \mathbf{G}B$ in $\mathcal{A}$ satisfying the $\mathbf{T}$-algebra homomorphism condition

$$h \circ \alpha = \mathbf{G}\varepsilon_B \circ \mathbf{T}h,$$

that is, diagram (4.10) should commute:

$$\begin{tikzcd}
\mathbf{T}A \arrow[r, "\mathbf{T}h"] \arrow[d, "\alpha"'] & \mathbf{TG}B \arrow[d, "\mathbf{G}\varepsilon_B"] \\
A \arrow[r, "h"'] & \mathbf{G}B.
\end{tikzcd} \tag{4.10}$$

Claim 4.2. *a morphism h in $\mathcal{A}$ satisfies (4.10) if and only if its adjunct via $\mathbf{F} \dashv \mathbf{G}$ coequalizes the pair of morphisms*

$$\begin{tikzcd}
\mathbf{FT}A \arrow[r, shift left, "\varepsilon_{\mathbf{F}A}"] \arrow[r, shift right, "\mathbf{F}\alpha"'] & \mathbf{F}A
\end{tikzcd}$$

Proof of claim 4.2. Denote the natural isomorphism of adjunction $\mathbf{F} \dashv \mathbf{G}$ by

$$\lfloor - \rfloor : \mathcal{B}(\mathbf{F}A, B) \cong \mathcal{A}(A, \mathbf{G}B) : \lceil - \rceil$$

we know from theorem 2.1 that the morphism h in $\mathcal{A}$ corresponds uniquely to its adjunct

$$\lceil h \rceil = \varepsilon_B \circ \mathbf{F}h : \mathbf{F}A \to B$$

in $\mathcal{B}$, so

$$\begin{array}{rrcl}
 & h \circ \alpha & = & \mathbf{G}\varepsilon_B \circ \mathbf{T}h \\
\Longleftrightarrow & \lceil h \circ \alpha \rceil & = & \lceil \mathbf{G}\varepsilon_B \circ \mathbf{T}h \rceil \\
\Longleftrightarrow & \varepsilon_B \circ \mathbf{F}(h \circ \alpha) & = & \varepsilon_B \circ \mathbf{F}(\mathbf{G}\varepsilon_B \circ \mathbf{T}h) \\
\Longleftrightarrow & (\varepsilon_B \circ \mathbf{F}h) \circ \mathbf{F}\alpha & = & (\varepsilon_B \circ \mathbf{FG}\varepsilon_B) \circ \mathbf{FT}h \\
\Longleftrightarrow & \lceil h \rceil \circ \mathbf{F}\alpha & = & (\varepsilon_B \circ \mathbf{FG}\varepsilon_B) \circ \mathbf{FGF}h \\
\Longleftrightarrow & \lceil h \rceil \circ \mathbf{F}\alpha & = & \varepsilon_B \circ \mathbf{FG}(\varepsilon_B \circ \mathbf{F}h) \\
\Longleftrightarrow & \lceil h \rceil \circ \mathbf{F}\alpha & = & \varepsilon_B \circ \mathbf{FG}\lceil h \rceil
\end{array}$$

By the naturality of ε, for any morphism $k : \mathbf{F}A \to B$ in $\mathcal{B}$, we have

$$\varepsilon_B \circ \mathbf{FG}k = k \circ \varepsilon_{\mathbf{F}A}.$$

In this case let $k = \lceil h \rceil = \varepsilon_B \circ \mathbf{F}h$, then

$$\lceil h \rceil \circ \mathbf{F}\alpha = \varepsilon_B \circ \mathbf{FG}\lceil h \rceil \iff \lceil h \rceil \circ \mathbf{F}\alpha = \lceil h \rceil \circ \varepsilon_{\mathbf{F}A}$$

i.e. h satisfies (4.10) if and only if $\lceil h \rceil$ coequalizes the pair $(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A})$. □

From claim 4.2, we have a natural bijection

$$\mathcal{A}^{\mathbf{T}}((A,\alpha), \mathbf{K}B) \cong \{\lceil h \rceil : \mathbf{F}A \to B \mid \lceil h \rceil \circ \mathbf{F}\alpha = \lceil h \rceil \circ \varepsilon_{\mathbf{F}A}\} \tag{4.11}$$

By hypothesis, $\mathcal{B}$ has coequalizers. Let

$$q_{(A,\alpha)} : \mathbf{F}A \to \mathrm{coeq}(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A})$$

denote the coequalizer of the above parallel pair. By the universal property of the coequalizer, every morphism $\lceil h \rceil : \mathbf{F}A \to B$ that coequalizes $(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A})$ factors uniquely through $q_{(A,\alpha)}$, so we have another natural bijection

$$\{\lceil h \rceil : \mathbf{F}A \to B \mid \lceil h \rceil \circ \mathbf{F}\alpha = \lceil h \rceil \circ \varepsilon_{\mathbf{F}A}\} \cong \mathcal{B}(\mathrm{coeq}(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A}), B). \tag{4.12}$$

Combining (4.11) and (4.12), we obtain a natural isomorphism

$$\mathcal{A}^{\mathbf{T}}((A,\alpha), \mathbf{K}B) \cong \mathcal{B}(\mathrm{coeq}(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A}), B). \tag{4.13}$$

Define $\mathbf{L}(A,\alpha) := \mathrm{coeq}(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A})$ via the coequalizer diagram

$$\begin{tikzcd} \mathbf{FGF}A \arrow[r, shift left, "\varepsilon_{\mathbf{F}A}"] \arrow[r, shift right, "\mathbf{F}\alpha"'] & \mathbf{F}A \arrow[r, "q_{(A,\alpha)}"] & \mathbf{L}(A,\alpha) \end{tikzcd} \tag{4.14}$$

The natural isomorphism (4.13) shows that $\mathbf{L}(A,\alpha)$ represents the functor

$$\mathcal{A}^{\mathbf{T}}((A,\alpha), \mathbf{K}-)$$

which extends uniquely to a functor $\mathbf{L} : \mathcal{A}^{\mathbf{T}} \to \mathcal{B}$ left adjoint to $\mathbf{K}$.

(2) Now assume $\mathbf{G}$ preserves coequalizers. We want to show the unit

$$\eta^{\mathbf{L}} : \mathbf{id}_{\mathcal{A}^{\mathbf{T}}} \implies \mathbf{KL}$$

is a natural isomorphism. Since $\mathbf{G}$ preserves coequalizers, applying $\mathbf{G}$ to the coequalizer diagram (4.14) yields another coequalizer in $\mathcal{A}$:

$$\begin{tikzcd} \mathbf{GFGF}A \arrow[r, shift left, "\mathbf{G}\varepsilon_{\mathbf{F}A}"] \arrow[r, shift right, "\mathbf{GF}\alpha"'] & \mathbf{GF}A \arrow[r, "\mathbf{G}q_{(A,\alpha)}"] & \mathbf{GL}(A,\alpha) \end{tikzcd} \tag{4.15}$$

i.e.

$$\begin{tikzcd} \mathbf{T}^2A \arrow[r, shift left, "\mu_A"] \arrow[r, shift right, "\mathbf{T}\alpha"'] & \mathbf{T}A \arrow[r, "\mathbf{G}q_{(A,\alpha)}"] & \mathbf{GL}(A,\alpha) \end{tikzcd} \tag{4.16}$$

But recall from example 4.3 that the $\mathbf{T}$-algebra structure map $\alpha : \mathbf{T}A \to A$ is also a coequalizer (Beck's coequalizer) of the same parallel pair $(\mathbf{T}\alpha, \mu_A)$, so there must exist a unique isomorphism $i : A \to \mathbf{GL}(A,\alpha)$ in $\mathcal{A}$ such that diagram (4.17) commutes:

$$\begin{tikzcd} \mathbf{T}A \arrow[r, "\alpha"] \arrow[dr, "\mathbf{G}q_{(A,\alpha)}"'] & A \arrow[d, dashed, "i"] \\ & \mathbf{GL}(A,\alpha) \end{tikzcd} \tag{4.17}$$

The unit $\eta^{\mathbf{L}}_{(A,\alpha)} : (A,\alpha) \to \mathbf{KL}(A,\alpha)$, is a morphism in $\mathcal{A}^{\mathbf{T}}$ thus a $\mathbf{T}$-algebra homomorphism whose underlying arrow in $\mathcal{A}$ is precisely this isomorphism i. Since a morphism of $\mathbf{T}$-algebras whose underlying arrow is an isomorphism is itself an isomorphism in $\mathcal{A}^{\mathbf{T}}$, it follows that $\eta^{\mathbf{L}}_{(A,\alpha)}$ is an isomorphism. As this holds for any algebra (A,α), the unit $\eta^{\mathbf{L}}$ is a natural isomorphism.

(3) Next, assume $\mathbf{G}$ is conservative. We want to show that the counit

$$\varepsilon^{\mathbf{L}} : \mathbf{LK} \implies \mathbf{id}_{\mathcal{B}}$$

is a natural isomorphism. Recall from (1) that $\mathbf{L}(A,\alpha)$ is defined as the coequalizer of the pair $(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A})$. Apply $\mathbf{L}$ to the $\mathbf{T}$-algebra $\mathbf{K}B = (\mathbf{G}B, \mathbf{G}\varepsilon_B)$, then the object $\mathbf{LK}B$ is the coequalizer of the pair

$$\begin{tikzcd} \mathbf{FTG}B \arrow[r, shift left, "\mathbf{FG}\varepsilon_B"] \arrow[r, shift right, "\varepsilon_{\mathbf{FG}B}"'] & \mathbf{FG}B \end{tikzcd} \tag{4.18}$$

with the coequalizer map denoted by $q_B : \mathbf{FG}B \to \mathbf{LK}B$. By the naturality of ε,

$$\varepsilon_B \circ \mathbf{FG}\varepsilon_B = \varepsilon_B \circ \varepsilon_{\mathbf{FG}B}.$$

i.e. ε_B coequalizes the pair in diagram (4.18). By universal property of coequalizers, the counit $\varepsilon^{\mathbf{L}}_B$ is the unique morphism making diagram (4.19) commute:

$$\begin{tikzcd}
\mathbf{FGFG}B \arrow[r, shift left, "\mathbf{FG}\varepsilon_B"] \arrow[r, shift right, "\varepsilon_{\mathbf{FG}B}"'] & \mathbf{FG}B \arrow[r, "q_B"] \arrow[dr, "\varepsilon_B"'] & \mathbf{LK}B \arrow[d, dashed, "\varepsilon^{\mathbf{L}}_B"] \\
 & & B
\end{tikzcd} \tag{4.19}$$

The commuting triangle gives the identity $\varepsilon^{\mathbf{L}}_B \circ q_B = \varepsilon_B$. Applying the functor $\mathbf{G}$ to this equation yields:

$$\mathbf{G}\varepsilon^{\mathbf{L}}_B \circ \mathbf{G}q_B = \mathbf{G}\varepsilon_B$$

Now we analyze the morphisms in $\mathcal{A}$. By (2), $\mathbf{G}$ preserves the above coequalizers. Hence

$$\begin{tikzcd}
\mathbf{GFGFG}B \arrow[r, shift left, "\mathbf{GFG}\varepsilon_B"] \arrow[r, shift right, "\mathbf{G}\varepsilon_{\mathbf{FG}B}"'] & \mathbf{GFG}B \arrow[r, "\mathbf{G}q_B"] & \mathbf{GLK}B
\end{tikzcd}$$

is a coequalizer in $\mathcal{A}$, i.e.

$$\begin{tikzcd}
\mathbf{T}^2\mathbf{G}B \arrow[r, shift left, "\mathbf{TG}\varepsilon_B"] \arrow[r, shift right, "\mu_{\mathbf{G}B}"'] & \mathbf{TG}B \arrow[r, "\mathbf{G}q_B"] & \mathbf{GLK}B
\end{tikzcd} \tag{4.20}$$

is a coequalizer in $\mathcal{A}$. By Beck's coequalizer, $\mathbf{G}\varepsilon_B : \mathbf{TG}B \to \mathbf{G}B$, which is the $\mathbf{T}$-algebra structure map on $\mathbf{K}B$, is also a coequalizer of diagram (4.20), so $\mathbf{G}\varepsilon^{\mathbf{L}}_B$ must be an isomorphism in $\mathcal{A}$. Since $\mathbf{G}$ reflects isomorphisms, it follows that $\varepsilon^{\mathbf{L}}_B$ is an isomorphism in $\mathcal{B}$.

By definition $\mathbf{K}$ is an equivalence of categories, i.e. $\mathbf{G}$ is monadic. □

Theorem 4.1 is not the original formulation in Beck's thesis, but a modernized version by Mac Lane [ML98]. In fact, we can make this theorem even more symmetric by replacing theorem 4.1(3) with a reflection of coequalizers. This is true by noticing that a functor reflecting all coequalizers is necessarily conservative.

Lemma 4.5. *If functor $\mathbf{G} : \mathcal{B} \to \mathcal{A}$ reflects all coequalizers, then G is conservative.*

Proof. Let $f : B \to B'$ be a morphism in $\mathcal{B}$ such that $\mathbf{G}f$ is an isomorphism in $\mathcal{A}$. Note that an isomorphism is the coequalizer of the identity pair, and specifically, $\mathbf{G}f$ is the coequalizer of the pair

$$(\mathrm{id}_{\mathbf{G}B}, \mathrm{id}_{\mathbf{G}B}) = (\mathbf{G}\,\mathrm{id}_B, \mathbf{G}\,\mathrm{id}_B)$$

Since $\mathbf{G}$ reflects coequalizers, the fact that $\mathbf{G}f$ is the coequalizer of $(\mathbf{G}\,\mathrm{id}_B, \mathbf{G}\,\mathrm{id}_B)$ implies that f must be the coequalizer of (id_B, id_B) in $\mathcal{B}$, which is necessarily an isomorphism: If $q = \mathrm{coeq}(\mathrm{id}, \mathrm{id})$, then $q \circ \mathrm{id} = q \circ \mathrm{id}$. For any h with $h \circ \mathrm{id} = h \circ \mathrm{id}$, there is a unique u such that $u \circ q = h$. Setting $h = \mathrm{id}$ gives a unique i such that $i \circ q = \mathrm{id}$. Since q is epi, $q \circ i = \mathrm{id}$ follows. Thus q is an iso. Therefore, f is an isomorphism. □

Theorem 4.2 (Weak Monadicity Theorem, $\textbf{W}\textsc{tt}^*$ [ML98, Exercise VI.7.2]). *Let* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ *be an adjunction, then* $\mathbf{G}$ *is monadic if*

(1) $\mathcal{B}$ *has all coequalizers.*

(2) $\mathbf{G}$ *preserves all coequalizers.*

(3) $\mathbf{G}$ *reflects all coequalizers.*

Proof. By theorem 4.1 and lemma 4.5. □

So far we can conclude that theorem 4.2 is stronger (at least not weaker) than theorem 4.1. We phrase this by saying:

$$\text{Functor } \mathbf{G} \text{ has } \textbf{W}\textsc{tt} \Rightarrow \mathbf{G} \text{ has } \textbf{W}\textsc{tt}^*.$$

In fact, the converse is also true, which asserts $\textbf{W}\textsc{tt}$ and $\textbf{W}\textsc{tt}^*$ are equivalent conditions.

Lemma 4.6. *A conservative functor reflects any coequalizers that it preserves.*

Proof. Let $\mathbf{F} : \mathcal{A} \to \mathcal{B}$ be a conservative functor. Suppose a parallel pair $f, g : X \rightrightarrows Y$ has a coequalizer $p : Y \to P$ in $\mathcal{A}$ and that $\mathbf{F}$ preserves it, so $\mathbf{F}p$ is the coequalizer of $(\mathbf{F}f, \mathbf{F}g)$ in $\mathcal{B}$. Let $q : Y \to Q$ be any other morphism in $\mathcal{A}$ such that $\mathbf{F}q = \mathrm{coeq}(\mathbf{F}f, \mathbf{F}g)$. Since both $\mathbf{F}p$ and $\mathbf{F}q$ are coequalizers of the same pair in $\mathcal{B}$, by the uniqueness of colimits, there exists a unique isomorphism $h : \mathbf{F}P \to \mathbf{F}Q$ such that $h \circ \mathbf{F}p = \mathbf{F}q$. Back in $\mathcal{A}$, since p is the coequalizer of (f, g), and q is a fork for this pair (because $\mathbf{F}$ is faithful as it reflects isomorphisms), the universal property of p guarantees that there exists a unique morphism $k : P \to Q$ such that $k \circ p = q$. Applying $\mathbf{F}$ to this last equation gives $\mathbf{F}k \circ \mathbf{F}p = \mathbf{F}q$. Comparing this with the equation from $\mathcal{B}$, we have $\mathbf{F}k \circ \mathbf{F}p = h \circ \mathbf{F}p$. Since $\mathbf{F}p$ is a coequalizer, it is an epimorphism, so we get $\mathbf{F}k = h$. Since h is an isomorphism, so is $\mathbf{F}k$. Because $\mathbf{F}$ is conservative, the morphism k must be an isomorphism in $\mathcal{A}$. Thus, $q = p \circ k$ is the composite of a coequalizer and an isomorphism, which means q is itself a coequalizer. This shows that $\mathbf{F}$ reflects the coequalizer. □

Proposition 4.1. *Functor* $\mathbf{G}$ *has* **Wtt** *if and only if it has* **Wtt***.

Proof. We have seen that $\mathbf{G}$ having **Wtt*** implies it has **Wtt** by lemma 4.5. Conversely, if $\mathbf{G}$ has **Wtt**, then by lemma 4.6, $\mathbf{G}$ reflects all coequalizers, so it has **Wtt***. □

4.2.2 Precise Monadicity Theorems

It is not until now that we arrive at the core result originally due to Beck. Theorem 4.1 is termed *weak* because it requires stronger preconditions than strictly necessary–$\mathbf{G}$ has to create *all* coequalizers– while the following theorem gives a precise characterization of monadicity by weakening the preconditions, also known as the *Precise Monadicity Theorem* or **Ptt**. Instead of requiring $\mathbf{G}$ to create all coequalizers, it only needs to create coequalizers of a special class of parallel pairs, called $\mathbf{G}$-*split pairs*.

Definition 4.9 (G-split pair)**.** For a functor $\mathbf{G} : \mathcal{B} \to \mathcal{A}$, a parallel pair $f, g : B_1 \rightrightarrows B_2$ in $\mathcal{B}$ is called a $\mathbf{G}$-*split pair* if the pair $(\mathbf{G}f, \mathbf{G}g)$ has a split coequalizer in $\mathcal{A}$.

Theorem 4.3 (Precise Monadicity Theorem, **Ptt** [BW00, Theorem 3.3.14])**.** *Let* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ *be an adjunction, then* $\mathbf{G}$ *is monadic if and only if the following conditions hold:*

(1) $\mathcal{B}$ *has coequalizers of all* $\mathbf{G}$*-split pairs.*

(2) $\mathbf{G}$ *is conservative.*

(3) $\mathbf{G}$ *preserves coequalizers of all* $\mathbf{G}$*-split pairs.*

Proof. ($\Rightarrow$) We show that these conditions are sufficient to satisfy the hypotheses of theorem 4.1, which in turn implies that $\mathbf{G}$ is monadic.

(1) asserts that $\mathcal{B}$ has coequalizers for all $\mathbf{G}$-split pairs. Let $\mathbf{L}$ be the quasi-inverse of $\mathbf{K}$, recall that $\mathbf{L}(A, \alpha)$ is defined as the coequalizer of $F\alpha$ and $\varepsilon_{\mathbf{F}A}$. Applying $\mathbf{G}$ to this pair yields the pair

$$(\mathbf{GF}\alpha, \mathbf{G}\varepsilon_{\mathbf{F}A}) = (\mathbf{T}\alpha, \mu_A)$$

This pair is part of the standard Beck's (split) coequalizer diagram involving $\mathbf{T}\eta_A$ and $\eta_{\mathbf{T}A}$. Therefore, this pair is $\mathbf{G}$-split, so by hypothesis (1), its coequalizer exists in $\mathcal{B}$ and is preserved by $\mathbf{G}$. Thus, the condition in theorem 4.1 for the existence of $\mathbf{L}$ is satisfied.

(2) satisfies theorem 4.1(3) for the counit of the adjunction $\mathbf{L} \dashv \mathbf{K}$ to be an isomorphism.

(3) asserts that $\mathbf{G}$ preserves these coequalizers, by theorem 4.1(2) the unit of the adjunction $\mathbf{L} \dashv \mathbf{K}$ is an isomorphism. One subtlety is that we are not applying $\mathbf{G}$ to all coequalizers in $\mathcal{B}$, but only to those of $\mathbf{G}$-split pairs, but the proves for both cases are identical.

All necessary conditions of **Wtt** are met, so $\mathbf{G}$ is monadic.

($\Leftarrow$) If $\mathbf{G}$ is monadic, there exists an equivalence $\mathcal{B} \simeq \mathcal{A}^{\mathbf{T}}$ under which $\mathbf{G}$ corresponds to the forgetful functor $\mathbf{G}^{\mathbf{T}} : \mathcal{A}^{\mathbf{T}} \to \mathcal{A}$.

(2) $\mathbf{G}^{\mathbf{T}}$ reflects isomorphisms. Let $h : (A, \alpha) \to (B, \beta)$ be a morphism of $\mathbf{T}$-algebras such that the underlying morphism h is an isomorphism in $\mathcal{A}$. We must show that h^{-1} is also a morphism of $\mathbf{T}$-algebras. Since h is a homomorphism, we have

$$h \circ \alpha = \beta \circ \mathbf{T}h$$

Since h is an isomorphism in $\mathcal{A}$, h^{-1} exists. Pre-composing with h^{-1} and post-composing with $(\mathbf{T}h)^{-1} = \mathbf{T}h^{-1}$ yields:

$$\alpha \circ \mathbf{T}(h^{-1}) = h^{-1} \circ \beta$$

This is precisely the condition for $h^{-1} : (B, \beta) \to (A, \alpha)$ to be a $\mathbf{T}$-algebra morphism. Thus, h is an isomorphism in $\mathcal{A}^{\mathbf{T}}$. Since the choice of h is arbitrary, $\mathbf{G}^{\mathbf{T}}$ reflects isomorphisms.

(1) $\mathbf{G}^{\mathbf{T}}$ has all coequalizers of $\mathbf{G}^{\mathbf{T}}$-split pairs and preserves them. Let $f, g : (A, \alpha) \rightrightarrows (B, \beta)$ be a pair of $\mathbf{T}$-algebra morphisms. Suppose the underlying pair f, g in $\mathcal{A}$ has a split coequalizer.

$$\begin{tikzcd}
A \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & B \arrow[l, bend left=50, "t"] \arrow[r, "q"] & Q \arrow[l, bend left=50, "s"]
\end{tikzcd}$$

Since split coequalizers are absolute, applying $\mathbf{T}$ yields a split coequalizer diagram in $\mathcal{A}$:

$$\begin{tikzcd}
\mathbf{T}A \arrow[r, shift left, "\mathbf{T}f"] \arrow[r, shift right, "\mathbf{T}g"'] & \mathbf{T}B \arrow[r, "\mathbf{T}q"] \arrow[l, bend left=50, "\mathbf{T}t"] & \mathbf{T}Q \arrow[l, bend left=50, "\mathbf{T}s"]
\end{tikzcd}$$

Consider the map $q \circ \beta : \mathbf{T}B \to Q$, it coequalizes $(\mathbf{T}f, \mathbf{T}g)$ because

$$(q \circ \beta) \circ \mathbf{T}f = q \circ f \circ \alpha = q \circ g \circ \alpha = (q \circ \beta) \circ \mathbf{T}g$$

Since $\mathbf{T}q$ is a coequalizer, by the universal property, there exists a unique morphism $\gamma : \mathbf{T}Q \to Q$ such that $\gamma \circ \mathbf{T}q = q \circ \beta$. The pair (Q, γ) forms a $\mathbf{T}$-algebra, because

1. Unit. We need $\gamma \circ \eta_Q = \mathrm{id}_Q$.

$$\gamma \circ \eta_Q \circ q = \gamma \circ \mathbf{T}q \circ \eta_B = q \circ \beta \circ \eta_B = q \circ \mathrm{id}_B = q$$

   Since q is a split epimorphism, we have $\gamma \circ \eta_Q = \mathrm{id}_Q$.

2. Associativity. Similarly, $\gamma \circ \mathbf{T}\gamma = \gamma \circ \mu_Q$ is true pre-composing with $\mathbf{T}^2 q$ (which is epimorphic).

Finally, the equation $\gamma \circ \mathbf{T}q = q \circ \beta$ states exactly that $q : (B, \beta) \to (Q, \gamma)$ is a homomorphism. Since $\mathbf{G}^{\mathbf{T}}$ creates limits, q is the coequalizer in $\mathcal{A}^{\mathbf{T}}$.

□

Again, we can apply lemma 4.3 to rewrite preconditions of **Ptt** into several equivalent forms as in theorems 4.4 and 4.5.

Theorem 4.4 (Precise Monadicity Theorem, **Ptt**$^{\diamond}$)**.** *Let $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ be an adjunction, then $\mathbf{G}$ is monadic if and only if the following conditions hold:*

(1) *$\mathcal{B}$ has coequalizers of all $\mathbf{G}$-split pairs.*

(2) *$\mathbf{G}$ preserves coequalizers of all $\mathbf{G}$-split pairs.*

(3) *$\mathbf{G}$ reflects coequalizers of all $\mathbf{G}$-split pairs.*

Proof. ($\Rightarrow$) The proof follows a similar strategy as in theorem 4.1. The idea is still to show that

(1) $\Rightarrow$ the comparison functor $\mathbf{K} : \mathcal{B} \to \mathcal{A}^{\mathbf{T}}$ admits a left adjoint $\mathbf{L}$.

(1)(2) $\Rightarrow$ the unit $\eta^{\mathbf{L}} : \mathbf{id}_{\mathcal{A}^{\mathbf{T}}} \Longrightarrow \mathbf{KL}$ is a natural isomorphism.

(1)(2)(3) $\Rightarrow$ the counit $\varepsilon^{\mathbf{L}} : \mathbf{LK} \Longrightarrow \mathbf{id}_{\mathcal{B}}$ is a natural isomorphism.

(1) We follow the construction of the left adjoint $\mathbf{L}$ from **Wtt**. Recall from theorem 4.1 that $\mathbf{L}(A, \alpha)$ is defined as the coequalizer of the pair $(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A})$ in $\mathcal{B}$. In **Wtt**, we assumed $\mathcal{B}$ had *all* coequalizers. Here, we simply check if this specific pair falls under hypothesis (1).

Applying $\mathbf{G}$ to the pair $(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A})$ yields the pair $(\mathbf{GF}\alpha, \mathbf{G}\varepsilon_{\mathbf{F}A}) = (\mathbf{T}\alpha, \mu_A)$. Again, it is part of the standard split coequalizer diagram for an algebra in $\mathcal{A}$:

$$\begin{tikzcd}
\mathbf{T}^2 A \arrow[r, shift left, "\mu_A"] \arrow[r, shift right, "\mathbf{T}\alpha"'] & \mathbf{T}A \arrow[l, bend left=60, "\eta_{\mathbf{T}A}"] \arrow[r, "\alpha"] & A \arrow[l, bend left=60, "\eta_A"]
\end{tikzcd} \tag{4.21}$$

By hypothesis (1), the coequalizer $q_{(A,\alpha)} : \mathbf{F}A \to \mathbf{L}(A, \alpha)$ exists in $\mathcal{B}$. The rest of the construction of $\mathbf{L}$ remains identical to **Wtt**.

(2) We must show the unit $\eta^{\mathbf{L}}$ is an isomorphism. In the proof of **Wtt**, this relied on $\mathbf{G}$ preserving the coequalizer of $(\mathbf{F}\alpha, \varepsilon_{\mathbf{F}A})$. As established in step (1), this pair is $\mathbf{G}$-split. Hypothesis (2) states that $\mathbf{G}$ preserves coequalizers of all $\mathbf{G}$-split pairs. Therefore, the image of the coequalizer diagram under $\mathbf{G}$:

$$\begin{tikzcd}
\mathbf{T}^2 A \arrow[r, shift left, "\mu_A"] \arrow[r, shift right, "\mathbf{T}\alpha"'] & \mathbf{T}A \arrow[r, "\mathbf{G}q_{(A,\alpha)}"] & \mathbf{GL}(A, \alpha)
\end{tikzcd}$$

is indeed a coequalizer in $\mathcal{A}$. From this point, the proof is verbatim identical to **Wtt**.

(3) Finally, we show the counit $\varepsilon^{\mathbf{L}}_B$ is an isomorphism. Recall from **Wtt** that $\mathbf{LK}B$ is defined as the coequalizer of the pair $(\mathbf{FG}\varepsilon_B, \varepsilon_{\mathbf{FG}B})$. In **Wtt**, we used the fact that $\mathbf{G}$ reflects isomorphisms. Here, we use the reflection of coequalizers.

First, observe that this pair is $\mathbf{G}$-split. Applying $\mathbf{G}$ gives $(\mathbf{GFG}\varepsilon_B, \mathbf{G}\varepsilon_{\mathbf{FG}B}) = (\mathbf{TG}\varepsilon_B, \mu_{\mathbf{G}B})$. This is the presentation of the free algebra on $\mathbf{G}B$, which admits a split coequalizer in $\mathcal{A}$ that is contractible to $\mathbf{G}B$ via $\mathbf{G}\varepsilon_B$. Since the image under $\mathbf{G}$ is a coequalizer diagram, and the pair is $\mathbf{G}$-split, hypothesis (3) implies that the original diagram in $\mathcal{B}$ is a coequalizer:

$$\begin{tikzcd}
\mathbf{FGFG}B \arrow[r, shift left] \arrow[r, shift right] & \mathbf{FG}B \arrow[r, "\varepsilon_B"] & B
\end{tikzcd} \tag{4.22}$$

However, by definition of $\mathbf{L}$, the object $\mathbf{LK}B$ is also the coequalizer of this exact pair. Since colimits are unique up to unique isomorphism, there exists a unique isomorphism $\mathbf{LK}B \cong B$. By the universal property, this isomorphism is exactly the counit $\varepsilon^{\mathbf{L}}_B$.

Thus, $\mathbf{G}$ is monadic.
$(\Leftarrow)$ When $\mathbf{G}$ is monadic, by theorem 4.3, theorem 4.4(1) is satisfied. Also theorem 4.3(2) ensures that $\mathbf{G}$ is conservative, and theorem 4.3(3) ensures that $\mathbf{G}$ preserves coequalizers of all $\mathbf{G}$-split pairs, so theorem 4.4(2) is satisfied. Finally, by lemma 4.6, $\mathbf{G}$ reflects coequalizers of all $\mathbf{G}$-split pairs, so theorem 4.4(3) is also satisfied. □

Theorem 4.4 shows in a direct way that what is precise about **Ptt** comparing to **Wtt**–it only requires $\mathcal{B}$ to have and $\mathbf{G}$ to preserve and to reflect coequalizers of a particular class of pairs, rather than all pairs. There is another equivalent formulation of **Ptt**, which is often referred to as the standard formulation of Beck's Precise Monadicity Theorem, presented in theorem 4.5.

Theorem 4.5 (Precise Monadicity Theorem, **Ptt***). *Let* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ *be an adjunction, then* $\mathbf{G}$ *is monadic if and only if* $\mathbf{G}$ creates *coequalizers of all* $\mathbf{G}$*-split pairs.*

Proof. By theorem 4.3, $\mathbf{G}$ is monadic if and only if theorem 4.3(2)(1)(3) are satisfied.
(**Ptt** $\Rightarrow$ **Ptt***)
By lemma 4.6, theorem 4.3(2) and theorem 4.3(3) implies that $\mathbf{G}$ reflects coequalizers of all $\mathbf{G}$-split pairs. By lemma 4.3, this implies that $\mathbf{G}$ creates coequalizers of all $\mathbf{G}$-split pairs.

(**Ptt*** $\Rightarrow$ **Ptt**$^\diamond$)
Note that $\mathcal{A}$ has coequalizers of all pairs $(\mathbf{G}f, \mathbf{G}g)$ where (f, g) is a $\mathbf{G}$-split pair in $\mathcal{B}$, because a pair (f, g) in $\mathcal{B}$ is $\mathbf{G}$-split if $(\mathbf{G}f, \mathbf{G}g)$ has a split coequalizer in $\mathcal{A}$, thus a coequalizer certainly exists. So by lemma 4.3, **Ptt*** is equivalent to the conjunction of theorem 4.3(1)(3) and that $\mathbf{G}$ reflects coequalizers of all $\mathbf{G}$-split pairs, which is exactly the condition of theorem 4.4.

(**Ptt**$^\diamond$ $\Leftrightarrow$ **Ptt**)
As shown in theorem 4.4, **Ptt**$^\diamond$ is equivalent to **Ptt**. □

Another precide monadicity theorem was presented on [ML98], which allows us to replace split coequalizers with absolute coequalizers.

Theorem 4.6 (Precise Monadicity Theorem, **Ptt**$^\bullet$). *Let* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ *be an adjunction, then* $\mathbf{G}$ *is monadic if and only if the following conditions hold:*

(1) $\mathcal{B}$ *has coequalizers for all* $\mathbf{G}$*-absolute pairs*[3].

[3] Similar to $\mathbf{G}$-split pairs, a pair f, g is called $\mathbf{G}$-absolute if $\mathbf{G}f, \mathbf{G}g$ admits absolute coequalizers.

(2) $\mathbf{G}$ *preserves coequalizers of all* $\mathbf{G}$*-absolute pairs.*

(3) $\mathbf{G}$ *reflects coequalizers of all* $\mathbf{G}$*-absolute pairs.*

Proof. One direction of the proof is straightforward.
($\Rightarrow$) By lemma 4.4 every split coequalizer is absolute, so every $\mathbf{G}$-split pair is a $\mathbf{G}$ absolute pair. If $\mathbf{G}$ has, preserves and reflects coequalizers of all $\mathbf{G}$-absolute pairs, it must also have, preserve and reflect coequalizers of all $\mathbf{G}$-split pairs, which by theorem 4.4 implies that $\mathbf{G}$ is monadic.

($\Leftarrow$) Assume $\mathbf{G}$ is monadic. As in the proof of theorem 4.4, we may (up to equivalence) identify $\mathcal{B}$ with the Eilenberg–Moore category $\mathcal{A}^{\mathbf{T}}$ of the induced monad $\mathbf{T} = \mathbf{GF}$, and identify $\mathbf{G}$ with the forgetful functor

$$\mathbf{U}^{\mathbf{T}} : \mathcal{A}^{\mathbf{T}} \to \mathcal{A}.$$

Under this identification, it suffices to show that $\mathbf{U}^{\mathbf{T}}$ creates coequalizers of all $\mathbf{U}^{\mathbf{T}}$-absolute pairs. Then lemma 4.3 applied to the class of all $\mathbf{U}^{\mathbf{T}}$-absolute pairs immediately yields (1)-(3). Let $f, g : (A, \alpha) \rightrightarrows (B, \beta)$ be a parallel pair in $\mathcal{A}^{\mathbf{T}}$ which is $\mathbf{U}^{\mathbf{T}}$-absolute. By definition, the underlying pair $\mathbf{U}f, \mathbf{U}g : A \rightrightarrows B$ admits an absolute coequalizer $q : B \to Q$ in $\mathcal{A}$. Since q is absolute, it is preserved by every functor, in particular by T. Thus $\mathbf{T}q : \mathbf{T}B \to \mathbf{T}Q$ is a coequalizer of $\mathbf{T}f, \mathbf{T}g$ in $\mathcal{A}$. Exactly as in the proof of theorem 4.4, we now lift q to a $\mathbf{T}$-algebra morphism. The composites $q \circ \beta : \mathbf{T}B \to Q$ coequalize $\mathbf{T}f, \mathbf{T}g$, and therefore by the universal property of the coequalizer $\mathbf{T}q$, there exists a unique map

$$\gamma : \mathbf{T}Q \to Q$$

such that

$$\gamma \circ \mathbf{T}q = q \circ \beta. \tag{4.23}$$

The verification that γ satisfies the $\mathbf{T}$-algebra axioms is identical to the calculation already carried out in the proof of theorem 4.4: precomposing both sides of the two required equations with the regular epis q and $\mathbf{T}^2 q$ reduces them to the corresponding equations for (B, β). Thus (Q, γ) is a $\mathbf{T}$-algebra, and (4.23) states exactly that

$$q : (B, \beta) \to (Q, \gamma)$$

is a $\mathbf{T}$-algebra morphism. To show that q is the coequalizer of f, g in $\mathcal{A}^{\mathbf{T}}$, let

$$h : (B, \beta) \to (C, \chi) \qquad \text{in } \mathcal{A}^{\mathbf{T}}$$

satisfy $h \circ f = h \circ g$. On underlying arrows in $\mathcal{A}$, this says $\mathbf{U}h \circ \mathbf{U}f = \mathbf{U}h \circ \mathbf{U}g$, so since q is a coequalizer of $\mathbf{U}f, \mathbf{U}g$, there is a unique map $k : Q \to C$ in $\mathcal{A}$ with $k \circ q = h$. The standard argument from theorem 4.4 now applies verbatim: precomposing $k \circ \gamma$ and $\chi \circ \mathbf{T}k : TQ \to C$ with the coequalizer $\mathbf{T}q$ and using (4.23) shows that both composites equal $h \circ \beta$. Since $\mathbf{T}q$ is an epi as a coequalizer, we conclude $k \circ \gamma = \chi \circ \mathbf{T}k$, so k is a $\mathbf{T}$-algebra morphism. Uniqueness follows from uniqueness of the underlying map in $\mathcal{A}$.

Thus q is the coequalizer of f, g in $\mathcal{A}^{\mathbf{T}}$ and its underlying arrow is the absolute coequalizer of $\mathbf{U}f, \mathbf{U}g$ in $\mathcal{A}$. Therefore $\mathbf{U}^{\mathbf{T}}$ creates coequalizers of all $\mathbf{U}^{\mathbf{T}}$-absolute pairs. Applying lemma 4.3 to this class yields that $\mathbf{U}^{\mathbf{T}}$ (hence $\mathbf{G}$) has, preserves, and reflects all coequalizers of $\mathbf{G}$-absolute pairs, establishing (1)-(3). □

4.2.3 Other Intermediate Monadicity Theorems

The power of precise monadicity theorems like **Ptt** or **Ptt**$^{\diamond}$ is that it provides necessary and sufficient conditions for monadicity. In practice, verifying its conditions directly can be complex. thus several simpler-to-verify but stronger sufficient conditions were also proposed, following the classical terminology of Mac Lane. We present them in order from the strongest to more specialized assumptions: the Crude, Vulgar, and Reflexive monadicity theorems.

Theorem 4.7 (Crude Monadicity Theorem, **Ctt**). *Let $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ be an adjunction, then $\mathbf{G}$ is monadic if it satisfies the following conditions:*

(1) *$\mathcal{B}$ has coequalizers for all $\mathbf{G}$-split pairs.*

(2) *$\mathbf{G}$ preserves all coequalizers.*

(3) *$\mathbf{G}$ reflects all coequalizers.*

Proof. It is clear from definition that **Ctt** has a stronger precondition than **Ptt**, as it requires $\mathbf{G}$ to preserve and reflect *all* coequalizers, not just those of $\mathbf{G}$-split pairs. Therefore, if $\mathbf{G}$ satisfies the conditions of **Ctt**, it also satisfies those of **Ptt**, and hence is monadic. □

Theorem 4.8 (Vulgar Monadicity Theorem, **Vtt**). *Let $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ be an adjunction, then $\mathbf{G}$ is monadic if it satisfies the following conditions:*

(1) *$\mathcal{B}$ has split coequalizers for $\mathbf{G}$-split pairs.*

(2) *$\mathbf{G}$ reflects coequalizers of all $\mathbf{G}$-split pairs.*

Proof. Just notice that split coequalizers are absolute colimits, so **G** automatically preserves them. Therefore the conditions of **Vtt** is stronger than those of **Ptt**, as it requires $\mathcal{B}$ to have split coequalizers for **G**-split pairs, instead of just coequalizers. Hence, if **G** satisfies the conditions of **Vtt**, it also satisfies those of **Ptt**, and thus is monadic. □

Theorem 4.9 (Reflexive Monadicity Theorem, **Rtt** [Rie17, Proposition 5.5.8])**.** *Let* $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$ *be an adjunction, then* **G** *is monadic if it satisfies the following conditions:*

(1) $\mathcal{B}$ *has all coequalizers of reflexive pairs.*

(2) **G** *is conservative.*

(3) **G** *preserves coequalizers of all reflexive.pairs.*

Proof. We prove by showing that these conditions implies **G** creates *reflective* **G**-split pairs. By hypothesis (1), since $\mathcal{B}$ has coequalizers of all reflexive pairs, it must have coequalizers of all reflexive **G**-split pairs. By hypothesis (3), **G** preserves coequalizers of all reflexive pairs, so it preserves coequalizers of all reflexive **G**-split pairs. By lemma 4.6, a concervative functor reflects any coequalizers that it preserves. Therefore **G** reflects coequalizers of all reflexive **G**-split pairs, so **G** creates coequalizers of all reflexive **G**-split pairs, which is stronger than the condition of **Ptt**. Hence, if **G** satisfies the conditions of **Rtt**, it also satisfies those of **Ptt**, and thus is monadic. □

So far, we have seen nine (theorems 4.1 to 4.9, essentially five) monadicity theorems in this section. and their relations can be summarized as a lattice in figure 4.1, where arrows indicate implication relations from the upper theorem to the lower one.

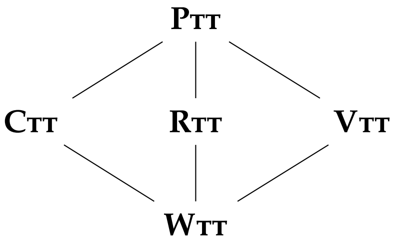

Figure 4.1: Implication relations between monadicity theorems.

Interestingly, in proofs of theorems 4.2, 4.4 and 4.9, the problem is often reduced to “finding the set of coequalizers that **G** needs to create” and maintaining corresponding preservation and reflection properties, thus naturally there arises a Galois correspondence between the class of coequalizers that $\mathbf{G}$ need to create and the strength of the monadicity theorem. For example, given an adjunction $\mathbf{F} \dashv \mathbf{G} : \mathcal{A} \to \mathcal{B}$, S be a class of all coequalizers in $\mathcal{B}$, $S^{\mathbf{G}}$ be the class of coequalizers for all $\mathbf{G}$-split pairs in $\mathcal{B}$, and $S_r^{\mathbf{G}}$ be the class of coequalizers for all reflexive $\mathbf{G}$-split pairs in $\mathcal{B}$. The following correspondence summarizes this idea:

$$\begin{tikzcd}
\textbf{\textsc{Ptt}} \arrow[d, dash, "\Downarrow"] & S \arrow[d, dash, "\rotatebox{90}{$\subseteq$}"] \\
\textbf{\textsc{Rtt}} \arrow[d, dash, "\Downarrow"] & S_r^{\mathbf{G}} \arrow[d, dash, "\rotatebox{90}{$\subseteq$}"] \\
\textbf{\textsc{Wtt}} & S^{\mathbf{G}}
\end{tikzcd}$$

This correspondence shows that the fewer coequalizers $\mathbf{G}$ needs to create, the stronger the monadicity theorem becomes–and the best we can do is the precise monadicity theorem, which requires $\mathbf{G}$ to create only coequalizers of $\mathbf{G}$-split pairs, while the worst case as shown in **Wtt** requires $\mathbf{G}$ to create coequalizers in the largest class S.

In fact, there are even more variants in the literature, including *Strict Monadicity Theorems*[Rie17][4], *Linton's Monadicity Theorem*[Lin66], *Duskin's Monadicity Theorem*[Dus06], *Relative Monadicity Theorems*[AM25] and many others, each focusing on different aspects of characterizations and thus having different application scenarios.

4.3 Examples of Monadic Adjunctions

Having established the hierarchy of monadicity theorems and their requisite conditions, we now turn to their practical application. To demonstrate the utility of these criteria, we examine a canonical case wherein the algebraic structure is clear. Consider our favorate example of the free-forgetful adjunction between **Grp** and **Set** in example 4.4.

Example 4.4 (Forgetful functor $\mathbf{G} : \mathbf{Grp} \to \mathbf{Set}$ is Monadic)**.** Let $\mathbf{G} : \mathbf{Grp} \to \mathbf{Set}$ be the forgetful functor from the category of groups to the category of sets. Then $\mathbf{G}$ is monadic. We verify the four conditions of the theorem 4.9

[4] Replacing *creation* with *strict creation*.

(1) **G** has a left adjoint **F**. This is satisfied by the *free group functor* $\mathbf{F} : \mathbf{Set} \to \mathbf{Grp}$, which sends any set X to the free group $\mathbf{F}X$ generated by the elements of X. Thus $\mathbf{F} \dashv \mathbf{G}$ is the canonical free–forgetful adjunction.

(2) **G** reflects isomorphisms. If $f : H \to K$ is a group homomorphism such that its underlying function $\mathbf{G}f : \mathbf{G}H \to \mathbf{G}K$ is a bijection, then f must be an isomorphism in **Grp**. Indeed, the set-theoretic inverse $f^{-1} : \mathbf{G}K \to \mathbf{G}H$ is easily checked to respect the group operation, hence it is a group homomorphism and serves as the inverse to f in **Grp**.

(3) **Grp** has coequalizers of all reflexive pairs. In fact, **Grp** is cocomplete and has coequalizers for all parallel pairs. For homomorphisms $f, g : G \rightrightarrows H$, the coequalizer is the quotient map $q : H \to H/N$, where N is the *normal closure* of the set $\{f(x)g(x)^{-1} \mid x \in G\}$.

(4) **G** preserves coequalizers of all reflexive pairs. Consider the coequalizer $q : H \to H/N$ from (3). Applying **G** yields the diagram of sets

$$\mathbf{G}H \xrightarrow{\mathbf{G}q} \mathbf{G}(H/N).$$

The set $\mathbf{G}(H/N)$ is the set of cosets of N in H, and the map $\mathbf{G}q$ sends $h \mapsto hN$. This is exactly the coequalizer of the set-maps $\mathbf{G}f, \mathbf{G}g : \mathbf{G}G \rightrightarrows \mathbf{G}H$ in **Set**, since the underlying equivalence relation on $\mathbf{G}H$ generated by $f(x) \sim g(x)$ is precisely the relation whose quotient is H/N. Thus **G** preserves coequalizers of reflexive pairs.

What is a non-example of a monadic functor? Consider the forgetful functor from topological spaces to sets.

Example 4.5 (The forgetful functor $\mathbf{G} : \mathbf{Top} \to \mathbf{Set}$ is *not* monadic)**.** Let $\mathbf{G} : \mathbf{Top} \to \mathbf{Set}$ be the underlying-set functor with left adjoint $\mathbf{F} : \mathbf{Set} \to \mathbf{Top}$ sending a set to the discrete topological space on that set. By theorem 4.3, if **G** were monadic, it would have to be conservative. But **G** does *not* reflect isomorphisms, because bijections are not necessarily homeomorphisms. To see this, choose a set X with at least two elements and let X_d be X with the discrete topology, X_i the same set with the indiscrete topology. The identity function

$$\mathrm{id}_X : (X_d \to X_i)$$

is a continuous bijection (every map into an indiscrete space is continuous). However, its inverse $\mathrm{id}_X : (X_i \to X_d)$ is *not* continuous unless X has at most one point.

Hence $\mathrm{id}_X : X_d \to X_i$ is *not* a homeomorphism, although $\mathbf{G}(\mathrm{id}_X)$ is a bijection. Therefore $\mathbf{G}$ fails to reflect isomorphisms.

What is the essence that makes the forgetful functor monadic? An intuition is that the forgetful functor would have to forget as much as an algebraic structure can be forgotten. In fact, **Top** is not an algebraic category over **Set**, thus **Top** cannot be a category of models of a finitary algebraic theory. Does it mean that monadic functors only arise from algebraic structures? Not necessarily. The free-forgetful adjunction between the category of *Stone Spaces* and **Set** is monadic, even though Stone spaces are not algebraic structures.

Chapter 5

Algebraic Theories

One of the most important and well-studied application of the monadicity theorem is to algebraic theories, where it provides a powerful framework for understanding the relationship between algebraic structures and their underlying sets. In this chapter, we will see why the Eilenberg-Moore category is often called the category of monad algebras. In particular, We will discuss present several examples of monadic adjunctions arising from categories of concrete algebraic structures, and then prove an equivalence between category of algebras and category of monad algebras over **Set** in section 5.1. We will finish the thesis with some following studies related to free constructions and monadicity, in universal algebra and theoretical computer science section 5.2.

5.1 Algebras are T-Algebras

At the beginning of chapter 3, we posited that monads could be intuitively understood as generalized algebraic structures. To formalize this intuition, we must examine how classical algebraic structures align with the categorical definition of algebras for a monad. In fact, many familiar algebraic structures can be characterized as algebras for certain monads. Moreover, the forgetful functors from categories of these algebraic structures to **Set** are often monadic.

We will firstly recall the definition of *single-sorted algebraic theories*, or *equational theories*.

Definition 5.1 (Equational Theory)**.** An *equational theory* or *single-sorted algebraic theory* is a pair $(\Sigma, \mathcal{E})$ where the *signature* $\Sigma = \{\mathcal{O}_n\}_{n\in\mathbb{N}}$ is a family of operation-symbol sets, and $\mathcal{E}$ is a set of equations between Σ-terms in finitely many variables. For each $\sigma \in \mathcal{O}_n$, we say it has *arity* n, written $\mathsf{ar}(\sigma) = n$.

For readers familiar with notions from Model Theory, an equational theory is essentially a first-order theory without relation symbols and only with function symbols of finite arity, where all axioms are equations between terms. In particular, all equations are in *Skolem normal form*, which are universally quantified equations between terms.

Definition 5.1 is not categorical, as $\Sigma, \mathcal{O}_n$ and $\mathcal{E}$ are just formal notions, However there is a beautiful categorical formalization of equational theories. First, we need to categorize the notion of signature using functors.

Definition 5.2 (Signature functor)**.** Given an equational theory $(\Sigma, \mathcal{E})$, the *signature endofunctor* $\mathbf{\Sigma} : \mathbf{Set} \to \mathbf{Set}$ is defined as a coproduct of functors $X \mapsto X^n$ for each operation symbol with arity n in Σ, explicitly,

$$\mathbf{\Sigma} X = \coprod_{n \in \mathbb{N}} \mathcal{O}_n \times X^n$$

From definition 5.2, to define functor $\mathbf{\Sigma}$ we implicitly require the underlying category to have finite coproducts and finite products. Since in classical algebraic theories we are often working over sets, we may as well assume the underlying category is $\mathbf{Set}$, which indeed has all small limits and colimits.

There is a known result from monadicity theorem that, given a signature functor $\mathbf{\Sigma} : \mathcal{A} \to \mathcal{A}$ over a category $\mathcal{A}$ with finite coproducts, the category of $\mathbf{\Sigma}$-algebras is monadic over $\mathcal{A}$ if and only if that the forgetful functor has a left adjoint. In particular, the following theorem characterizes this.

Theorem 5.1. [Awo06] *Let $\mathcal{A}$ have finite coproducts and $\mathbf{\Sigma} : \mathcal{A} \to \mathcal{A}$ be an endofunctor. The following are equivalent:*

(1) *There exists an monad $\mathbf{T}$ on $\mathbf{\Sigma}$ such that*

$$\mathcal{A}^{\mathbf{T}} \simeq \mathbf{\Sigma}\text{-}\mathbf{Alg}.$$

(2) *The forgetful $\mathbf{U} : \mathbf{\Sigma}\text{-}\mathbf{Alg} \to \mathcal{A}$ has a left adjoint.*

(3) *For each $A \in \operatorname{Ob} \mathcal{A}$, the endofunctor $\mathbf{\Sigma}_A(-) = A + \mathbf{\Sigma}(-)$ has an initial algebra.*

Proof. I'll prove by showing (1) ⇒ (2) ⇒ (3) ⇒ (2) ⇒ (1).

(1) ⇒ (2) Consider the comparison functor $\mathbf{K} : \mathbf{\Sigma}\text{-}\mathbf{Alg} \to \mathcal{A}^{\mathbf{T}}$, since $\mathbf{K}$ is an equivalence, it has a quasi-inverse $\mathbf{J}$, and in particular, $\mathbf{J} \dashv \mathbf{K}$. Consider the adjunction $\mathbf{F}^{\mathbf{T}} \dashv \mathbf{U}^{\mathbf{T}}$ between $\mathcal{A}$ and the Eilenberg-Moore category $\mathcal{A}^{\mathbf{T}}$. By lemma 2.2 adjunctions are composable, we have $\mathbf{J} \circ \mathbf{F}^{\mathbf{T}} \dashv \mathbf{G}^{\mathbf{T}} \circ \mathbf{K} \cong \mathbf{U}$, by lemma 2.1 left adjoints are unique up to isomorphism, so $\mathbf{J} \circ \mathbf{F}^{\mathbf{T}}$ is left adjoint to $\mathbf{U}$.

(2) ⇒ (3) Suppose that $\mathbf{U}$ has a left adjoint $\mathbf{F} : \mathcal{A} \to \mathbf{\Sigma}\text{-}\mathbf{Alg}$ and consider the endofunctor $\mathbf{\Sigma}_A X = A + \mathbf{\Sigma} X$. Given a $\mathbf{\Sigma}_A$-algebra $(X, \gamma : A + \mathbf{\Sigma} X \to X)$, by the universal property of coproduct, γ uniquely decomposes as $[\alpha, \beta]$, where

$$\alpha : A \to X \qquad \beta : \mathbf{\Sigma} X \to X$$

and conversely any pair (α, β) gives a unique γ. Thus, a $\mathbf{\Sigma}_A$-algebra is exactly the same data as

a $\mathbf{\Sigma}$ algebra (X, β) and a morphism $\alpha : A \to \mathbf{U}(X, \beta) = X$

For each A in $\mathcal{A}$, denote $\mathbf{F}A$ by

$$\mathbf{F}A = (\mathbf{\Sigma}^* A, \mu_A : \mathbf{\Sigma}\mathbf{\Sigma}^* A \to \mathbf{\Sigma}^* A)$$

where $\mathbf{\Sigma}^* = \mathbf{U}\mathbf{F}$, and consider the unit $\eta_A : A \to \mathbf{\Sigma}^* A$ of the adjunction $\mathbf{F} \dashv \mathbf{U}$.

Claim 5.1. *$I_A = (\mathbf{\Sigma}^* A, [\eta_A, \mu_A] : A + \mathbf{\Sigma}\mathbf{\Sigma}^* A \to \mathbf{\Sigma}^* A)$ is the initial $\mathbf{\Sigma}_A$-algebra.*

Proof of claim 5.1. To see the universal property, let $(X, [\alpha, \beta])$ be any $\mathbf{\Sigma}_A$-algebra, where $\alpha : A \to \mathbf{\Sigma}^* A$ and $\beta : \mathbf{\Sigma}\mathbf{\Sigma}^* A \to \mathbf{\Sigma}^* A$. By definition 2.17 of the adjunction $\mathbf{F} \dashv \mathbf{U}$, there exists a unique $\mathbf{\Sigma}$-algebra homomorphism $h : \mathbf{F}A \to X$ such that

$$\mathbf{U}h \circ \eta_A = \alpha$$

Using the universal property of coproducts,

$$[\alpha, \beta]\,(\mathrm{id}_A + \mathbf{\Sigma} h) = [\alpha, \beta \circ \mathbf{\Sigma} h] \qquad h \circ [\eta_A, \mu_A] = [h \circ \eta_A, h \circ \mu_A]$$

But meanwhile,

1. $h \circ \eta_A = \mathbf{U}h \circ \eta_A = \alpha$, by the universal property of adjunction, and

2. $h \circ \mu_A = \beta \circ \mathbf{\Sigma} h$, because $h : (\mathbf{\Sigma}^* A, \mu_A) \to (X, \beta)$ is a $\mathbf{\Sigma}$-algebra homomorphism.

so we have

$$[\alpha, \beta]\,(\mathrm{id}_A + \mathbf{\Sigma} h) = h \circ [\eta_A, \mu_A]$$

implying that the following diagram commutes

$$\begin{tikzcd}
A + \mathbf{\Sigma}\mathbf{\Sigma}^* A \arrow[r, "\mathrm{id}_A + \mathbf{\Sigma} h"] \arrow[d, "{[\eta_A, \mu_A]}"'] & A + \mathbf{\Sigma} X \arrow[d, "{[\alpha, \beta]}"] \\
\mathbf{\Sigma}^* A \arrow[r, "h"'] & X
\end{tikzcd}$$

so there is always a morphism from I_A to any arbitrary $\mathbf{\Sigma}_A$-algebra $(X, [\alpha, \beta])$. To see the uniqueness, let $k : \mathbf{\Sigma}^* A \to X$ be any $\mathbf{\Sigma}_A$-algebra homomorphism, then similarly we have

$$[\alpha, \beta]\,(\mathrm{id}_A + \mathbf{\Sigma} k) = k \circ [\eta_A, \mu_A]$$

precomposing the coproduct injections gives

$$k \circ \eta_A = \alpha$$

so k is a $\mathbf{\Sigma}$-algebra homomorphism with $\mathbf{U} k \circ \eta_A = \alpha$. By the adjunction, the $\mathbf{\Sigma}$-algebra homomorphism with this property is unique, hence $k = h$. This shows the initiality of I_A, i.e. I_A is the initial $\mathbf{\Sigma}_A$-algebra. □

(3) ⇒ (2) Assume for each $A \in \operatorname{Ob} \mathcal{A}$, the endofunctor $\mathbf{\Sigma}_A(-) = A + \mathbf{\Sigma}(-)$ has an initial algebra

$$(I_A, \iota_A : A + \mathbf{\Sigma} I_A \to I_A)$$

By the coproduct property, write $\iota_A = [\eta_A, \mu_A]$, with

$$\eta_A : A \to I_A \qquad \mu_A : \mathbf{\Sigma} I_A \to I_A$$

so (I_A, μ_A) is a $\mathbf{\Sigma}$-algebra, and $\eta_A : A \to I_A$ is a chosen morphism into its underlying object. Define $\mathbf{F} : \mathcal{A} \to \mathbf{\Sigma}\text{-}\mathbf{Alg}$ on objects by

$$\mathbf{F} A = (I_A, \mu_A)$$

and on morphisms as follows. Let $f : A \to B$ be any morphism in $\mathcal{A}$, consider the $\mathbf{\Sigma}$-algebra $\mathbf{F}B = (I_B, \mu_B)$ together with the morphism $\alpha := \eta_B \circ f : A \to I_B$. This forms a $\mathbf{\Sigma}_A$-algebra

$$(I_B, [\eta_B \circ f, \mu_B] : A + \mathbf{\Sigma} I_B \to I_B)$$

By initiality of $(I_A, [\eta_A, \mu_A])$, there exists a unique morphism $\mathbf{F}f : (I_A, \mu_A) \to (I_B, \mu_B)$ such that the following diagram commutes

$$\begin{tikzcd}
A + \mathbf{\Sigma} I_A \arrow[r, dashed, "\mathrm{id}_A + \mathbf{\Sigma F}f"] \arrow[d, "{[\eta_A, \mu_A]}"'] & A + \mathbf{\Sigma} I_B \arrow[d, "{[\eta_B \circ f, \mu_B]}"] \\
I_A \arrow[r, dashed, "\mathbf{F}f"'] & I_B
\end{tikzcd}$$

The uniqueness of $\mathbf{F}f$ guarantees functoriality of $\mathbf{F}$, i.e.

$$\mathbf{F}\mathrm{id}_A = \mathrm{id}_{\mathbf{F}A} \qquad \mathbf{F}(g \circ f) = \mathbf{F}g \circ \mathbf{F}f$$

so $\mathbf{F}$ is a well-defined functor from $\mathcal{A}$ to $\mathbf{\Sigma}$-$\mathbf{Alg}$.

Claim 5.2. $\mathbf{F} \dashv \mathbf{U}$.

Proof of claim 5.2. Define function $\Phi : \mathbf{\Sigma}\text{-}\mathbf{Alg}(\mathbf{F}A, (X, \beta)) \to \mathcal{A}(A, X)$ by mapping a $\mathbf{\Sigma}$-algebra homomorphism $h : (I_A, \mu_A) \to (X, \beta)$ to the composite

$$\Phi(h) = h \circ \eta_A : A \to X$$

Define function $\Psi : \mathcal{A}(A, X) \to \mathbf{\Sigma}\text{-}\mathbf{Alg}(\mathbf{F}A, (X, \beta))$. Consider a morphism $\alpha : A \to X$, then $(X, [\alpha, \beta])$ forms a $\mathbf{\Sigma}_A$-algebra, by initiality of $(I_A, [\eta_A, \mu_A])$, there exists a unique $\mathbf{\Sigma}$-algebra homomorphism

$$\overline{\alpha} : (I_A, \mu_A) \to (X, \beta)$$

such that $\overline{\alpha} \circ \eta_A = \alpha$. Define $\Psi(\alpha) = \overline{\alpha}$.

1. $\Phi(\Psi(\alpha)) = \Phi(\overline{\alpha}) = \overline{\alpha} \circ \eta_A = \alpha$, so $\Phi \circ \Psi = \mathrm{id}$,
2. Let $h : (I_A, \mu_A) \to (X, \beta)$ be a $\mathbf{\Sigma}$-algebra homomorphism, set $\alpha = h \circ \eta_A$,then $\Psi(\Phi(h)) = \overline{\alpha}$, which is the unique $\mathbf{\Sigma}$-algebra homomorphism such that $\overline{\alpha} \circ \eta_A = \alpha = h \circ \eta_A$, by uniqueness, $\overline{\alpha} = h$, so $\Psi \circ \Phi = \mathrm{id}$.

The bijection is natural because both Φ and Ψ are defined using only composition with $\mathbf{\Sigma}$-algebra morphisms, which is functorial. This shows that $\mathbf{F} \dashv \mathbf{U}$. □

(2) ⇒ (1) This is true by applying theorem 4.5 **Ptt***, that $\mathbf{U}$ is monadic if it has a left adjoint and creates $\mathbf{U}$-split coequalizers. Let $\mathbf{F}$ be the left adjoint of $\mathbf{U}$, (X, β) and (Y, γ) be $\mathbf{\Sigma}$-algebras, and let $f, g : (X, \beta) \rightrightarrows (Y, \gamma)$ be a pair of parallel morphisms, so we have a parallel of morphisms $f, g : X \rightrightarrows Y$ for the underlying objects with the additional usual homomorphism condition

$$f \circ \beta = \gamma \circ \mathbf{\Sigma} f \qquad g \circ \beta = \gamma \circ \mathbf{\Sigma} g$$

Assume this pair is $\mathbf{U}$-split. That means that in $\mathcal{A}$ we have a split coequalizer diagram:

$$\begin{tikzcd}
X \arrow[r, shift left, "\mathbf{U}f=f"] \arrow[r, shift right, "\mathbf{U}g=g"'] & Y \arrow[l, bend left=50, "s"] \arrow[r, "q"] & A \arrow[l, bend left=50, "t"]
\end{tikzcd}$$

with maps $s : Y \to X$ and $t : A \to Y$ satisfying

$$q \circ f = q \circ g \qquad q \circ s = \mathrm{id}_A \qquad f \circ t = \mathrm{id}_Y \qquad g \circ t = s \circ q$$

Define $\alpha : \mathbf{\Sigma} A \to A$ via the composite

$$\begin{tikzcd}
\mathbf{\Sigma} A \arrow[r, "\mathbf{\Sigma} s"] \arrow[rrr, bend right=20, two heads, "\alpha"'] & \mathbf{\Sigma} Y \arrow[r, "\gamma"] & Y \arrow[r, two heads, "q"] & A.
\end{tikzcd}$$

Observe that $q : (Y, \gamma) \to (A, \alpha)$ is a well-defined $\mathbf{\Sigma}$-algebra homomorphism, because

$$\begin{aligned}
\alpha \circ \mathbf{\Sigma} q &= (q \circ \gamma \mathbf{\Sigma} s) \circ \mathbf{\Sigma} q = q \circ \gamma \circ \mathbf{\Sigma}(s \circ q) \\
&= q \circ \gamma \circ \mathbf{\Sigma}(g \circ t) = q \circ \gamma \circ \mathbf{\Sigma} g \circ \mathbf{\Sigma} t \\
&= q \circ (\gamma \circ \mathbf{\Sigma} g) \circ \mathbf{\Sigma} t = q \circ g \circ \beta \circ \mathbf{\Sigma} t \\
&= (q \circ g) \circ \beta \circ \mathbf{\Sigma} t = q \circ f \circ \beta \circ \mathbf{\Sigma} t \\
&= q \circ (f \circ \beta) \circ \mathbf{\Sigma} t = q \circ \gamma \circ \mathbf{\Sigma} f \circ \mathbf{\Sigma} t \\
&= q \circ \gamma \circ \mathbf{\Sigma}(f \circ t) = q \circ \gamma \circ \mathbf{\Sigma} \mathrm{id}_Y \\
&= q \circ \gamma
\end{aligned}$$

Meanwhile, q is a coequalizer of f, g in $\mathbf{\Sigma}$-**Alg**, because consider $k : (Y, \gamma) \to (Z, \delta)$ be any $\mathbf{\Sigma}$-algebra homomorphism that coequalizes f and g in $\mathbf{\Sigma}$-**Alg**, i.e.

$$k \circ f = k \circ g$$

applying $\mathbf{U}$ to both sides, we have $\mathbf{U}k$ coequalizes $\mathbf{U}f = f$ and $\mathbf{U}g = g$, i.e.

$$\mathbf{U}k \circ \mathbf{U}f = \mathbf{U}k \circ \mathbf{U}g \implies \mathbf{U}k \circ f = \mathbf{U}k \circ g$$

Since $q : Y \to A$ is a coequalizer in $\mathcal{A}$, there exists a unique morphism $\ell : A \to \mathbf{U}Z$ in $\mathcal{A}$ such that $\ell \circ q = \mathbf{U}k$, i.e. the following diagram in $\mathcal{A}$ commutes.

$$\begin{tikzcd}
X \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & Y \arrow[r, "q"] \arrow[dr, "\mathbf{U}k"'] & A \arrow[d, dashed, "\ell"] \\
 & & Z
\end{tikzcd} \tag{5.1}$$

Claim 5.3. *ℓ is a $\mathbf{\Sigma}$-algebra homomorphism, because*

$$\begin{aligned}
\ell \circ \alpha = \ell \circ q \circ \gamma \circ \mathbf{\Sigma}s &= (\ell \circ q) \circ \gamma \circ \mathbf{\Sigma}s \\
&= \mathbf{U}k \circ \gamma \circ \mathbf{\Sigma}s = (\mathbf{U}k \circ \gamma) \circ \mathbf{\Sigma}s \\
&= (\delta \circ \mathbf{\Sigma}(\mathbf{U}k)) \circ \mathbf{\Sigma}s && (k \text{ is a homomorphism}) \\
&= \delta \circ \mathbf{\Sigma}(\mathbf{U}k \circ s) = \delta \circ \mathbf{\Sigma}\ell
\end{aligned}$$

i.e. the following diagram commutes

$$\begin{tikzcd}
\mathbf{\Sigma}X \arrow[r, "\mathbf{\Sigma}\ell"] \arrow[d, "\alpha"'] & \mathbf{\Sigma}Z \arrow[d, "\delta"] \\
A \arrow[r, "\ell"'] & Z
\end{tikzcd}$$

*which means that $\ell : (A, \alpha) \to (Z, \delta)$ is indeed a $\mathbf{\Sigma}$-algebra homomorphism. Since $\ell \circ q = \mathbf{U}k = k$ as $\mathbf{\Sigma}$-algebra homomorphisms, k factors through q in $\mathbf{\Sigma}$-***Alg**.

To see the uniqueness of ℓ, suppose $\ell_1, \ell_2 : (A, \alpha) \to (Z, \delta)$ are a parallel pair of $\mathbf{\Sigma}$-homomorphisms that are equalized q, i.e. $\ell_1 \circ q = \ell_2 \circ q$ as maps $(Y, \gamma) \to (Z, \delta)$. Applying $\mathbf{U}$, we get $\mathbf{U}\ell_1$ and $\mathbf{U}\ell_2$ that are equalized by q in $\mathbf{\Sigma}$-**Alg**, i.e.

$$\mathbf{U}\ell_1 \circ q = \mathbf{U}\ell_2 \circ q$$

Since q is a coequalizer in $\mathcal{A}$, it's an epimorphism, so $\mathbf{U}\ell_1 = \mathbf{U}\ell_2$, but since $\mathbf{U}$ is faithful, $\ell_1 = \ell_2$. So ℓ is a coequalizer of f, g in $\mathbf{\Sigma}$-**Alg**. Since the choice of f and g is arbitrary, by definition $\mathbf{U}$ creates $\mathbf{U}$-split coequalizers. By theorem 4.5, $\mathbf{T} = \mathbf{UF}$ satisfies $\mathcal{A}^{\mathbf{T}} \simeq \mathbf{\Sigma}$-**Alg**, which is exactly the statement in (1).

□

Before moving on, let us take a moment to inspect the free functor $\mathbf{F} : \mathcal{A} \to \mathbf{\Sigma}$-**Alg**. In theorem 5.1(3) ⇒ (2), we constructed $\mathbf{F}$ as follows

$$\mathbf{F}X = \mu Y.X + \mathbf{\Sigma}Y$$

which is the initial algebra of the endofunctor $\mathbf{\Sigma}_X(-) = X + \mathbf{\Sigma}(-)$. Unfolding this definition, $\mathbf{F}X = (\mathbf{\Sigma}^* X, \alpha)$ could be understood as the algebra whose underlying set $\mathbf{\Sigma}^* X$ of Σ-terms is the *smallest* set such that

1. Every element in X is a Σ-term, and
2. For every operation symbol $\sigma \in \mathcal{O}_n$ and elements $x_1, \ldots, x_n$ in $\mathbf{\Sigma}^* X$, the formal expression $\sigma(x_1, \ldots, x_n)$ is also a Σ-term.

and the reason we say *smallest* is exaclty due to Lambek's Lemma (lemma 3.1), which asserts that the structure map $\alpha : X + \mathbf{\Sigma}\mathbf{\Sigma}^* X \to \mathbf{\Sigma}^* X$ is an isomorphism. Moreover, the free-forgetful adjunction $\mathbf{F} \dashv \mathbf{U}$ induces an universal property of $\mathbf{F}X$

Proposition 5.1 (Universal property of free $\mathbf{\Sigma}$-algebras). *Let $\mathbf{\Sigma} : \mathbf{Set} \to \mathbf{Set}$ be the signature functor for an equational theory $(\Sigma, \mathcal{E})$. Then the free $\mathbf{\Sigma}$-algebra $\mathbf{F}X = (\mathbf{\Sigma}^* X, \alpha)$ on a set X satisfies the following universal property: for any $\mathbf{\Sigma}$-algebra (Y, β) and any interpretation function $[\![-]\!] : X \to Y$, there exists a unique $\mathbf{\Sigma}$-algebra homomorphism $\overline{[\![-]\!]} : \mathbf{F}X \to (Y, \beta)$ such that the following diagram commutes*

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[dr, "{[\![-]\!]}"'] & \mathbf{\Sigma}^* X \arrow[d, "\overline{[\![-]\!]}"] \\
 & Y
\end{tikzcd}$$

where $\eta_X : X \to \mathbf{\Sigma}^* X$ is the unit of the adjunction.

That is great! Theorem 5.1 tells us once given an algebraic theory $(\Sigma, \mathcal{E})$, we can just routinely construct the signature functor $\mathbf{\Sigma} : \mathbf{Set} \to \mathbf{Set}$, and the problem of inspecting what does $\mathbf{\Sigma}$-algebras looks like would be reduced to checking whether the forgetful functor $\mathbf{U} : \mathbf{\Sigma}\text{-}\mathbf{Alg} \to \mathbf{Set}$ has a left adjoint–because if it has one, there is an equivalence between the category of $\mathbf{\Sigma}$-algebras and the Eilenberg-Moore category $\mathbf{Set}^{\mathbf{T}}$ for the induced monad $\mathbf{T}$. In fact, functors defined in definition 5.2 are called *polynomial functors* for obvious reasons, and $\mathbf{\Sigma}_A$, which is a coproduct of identity functor and a polynomial functor, is also a polynomial functor. It is a well-known fact that polynomial functors are finitary[1] on $\mathbf{Set}$ always have initial algebras, thus we get an equivalence $\mathbf{\Sigma}\text{-}\mathbf{Alg} \simeq \mathbf{Set}^{\mathbf{T}}$ for free!

But there is one thing we left out–how to enforce the equations $\mathcal{E}$ in the theory $(\Sigma, \mathcal{E})$? Consider $\mathsf{T}_1 = (\Sigma, \mathcal{E}_1)$, the theory of groups, where Σ consists of a binary operation symbol $\cdot$, a unary operation symbol $(-)^{-1}$ and a nullary operation symbol e, and $\mathcal{E}_1$ consists of the usual group axioms.

$$\begin{aligned}(x \cdot y) \cdot z &= x \cdot (y \cdot z)\\ e \cdot x &= x\\ x \cdot e &= x\\ x \cdot x^{-1} &= e\\ x^{-1} \cdot x &= e\end{aligned}$$

The signature functor $\mathbf{\Sigma} : \mathbf{Set} \to \mathbf{Set}$ is given by

$$\mathbf{\Sigma} X = X \times X + X + 1$$

But also consider the theory $\mathsf{T}_2 = (\Sigma, \mathcal{E}_2)$ of abelian groups, where Σ is the same as before, but

$$\mathcal{E}_2 = \mathcal{E}_1 \cup \{x \cdot y = y \cdot x\}$$

without changing the signature functor $\mathbf{\Sigma}$. Our equivalence $\mathbf{\Sigma}\text{-}\mathbf{Alg} \simeq \mathbf{Set}^{\mathbf{T}}$ cannot distinguish between these two theories yet, without considering the equations $\mathcal{E}_1$ and $\mathcal{E}_2$.

To see this, we need to extend $\mathbf{\Sigma}\text{-}\mathbf{Alg}$ to more refined categories called *equational categories*, which are categories consisting of $\mathbf{\Sigma}$-algebras that satisfy the equations in $\mathcal{E}$. To define this we will firstly need to explain what does it mean to *satisfy* an equation.

[1] preserve filtered colimits, skipped in this thesis

Definition 5.3 (Equation)**.** Given an equational theory $(\Sigma, \mathcal{E})$, an *equation* in $\mathcal{E}$ is a pair of Σ-terms (t_1, t_2) in n variables, written as

$$\{x_1, \dots, x_n\} \mid t_1 = t_2$$

for some $n \in \mathbb{N}$. A $\mathbf{\Sigma}$-algebra $(X, \alpha : \mathbf{\Sigma} X \to X)$ *satisfies* the equation $t_1 = t_2$ if for variable interpretation $[\![-]\!] : \{x_1, \dots, x_n\} \to X$, the induced $\mathbf{\Sigma}$-algebra homomorphism $\overline{[\![-]\!]} : \mathbf{F}\{x_1, \dots, x_n\} \to (X, \alpha)$ satisfies

$$\overline{[\![t_1]\!]} = \overline{[\![t_2]\!]}$$

Definition 5.4 (Equational Category)**.** Let $(\Sigma, \mathcal{E})$ be an equational theory with signature functor $\mathbf{\Sigma} : \mathbf{Set} \to \mathbf{Set}$. The *equational category* of this theory is the full subcategory of $\mathbf{\Sigma}$-$\mathbf{Alg}$ formed by all small $(\mathbf{\Sigma}, \mathcal{E})$-algebras, which are $\mathbf{\Sigma}$-algebras that satisfy all equations in $\mathcal{E}$. We denote such a category by $(\mathbf{\Sigma}, \mathcal{E})$-$\mathbf{Alg}$.

In theorem 5.1, we have actually only seen an equivalence between the three conditions, that is, $\mathcal{A}^{\mathbf{T}} \simeq \mathbf{\Sigma}$-$\mathbf{Alg}$ under certain conditions. The following theorem proves the free-forgetful adjunction between $\mathbf{Set}$ and the equational category $(\mathbf{\Sigma}, \mathcal{E})$-$\mathbf{Alg}$ is indeed monadic.

Theorem 5.2. *The forgetful functor* $\mathbf{U} : (\mathbf{\Sigma}, \mathcal{E})$-$\mathbf{Alg} \to \mathbf{Set}$ *is monadic.*

Proof. Let $\mathbf{U} : (\mathbf{\Sigma}, \mathcal{E})$-$\mathbf{Alg} \to \mathbf{Set}$ be the forgetful functor. Recall from theorem 5.1 we constructed the left adjoint $\mathbf{F} : \mathbf{Set} \to \mathbf{\Sigma}$-$\mathbf{Alg} \cong (\mathbf{\Sigma}, \emptyset)$-$\mathbf{Alg}$ via

$$\mathbf{F}X = (\mathbf{\Sigma}^* X, \alpha : \mathbf{\Sigma}\mathbf{\Sigma}^* X \to \mathbf{\Sigma}^* X)$$

It would be a straightforward check to see that here $\mathbf{U}$ also has a left adjoint $\mathbf{F} : \mathbf{Set} \to (\mathbf{\Sigma}, \mathcal{E})$-$\mathbf{Alg}$ defined by

$$\mathbf{F}X = (\tilde{\mathbf{\Sigma}}^* X, \alpha : \mathbf{\Sigma}\tilde{\mathbf{\Sigma}}^* X \to \tilde{\mathbf{\Sigma}}^* X)$$

and $\tilde{\mathbf{\Sigma}}^* X$ is the quotient of $\mathbf{\Sigma}^* X$ by the smallest congruence relation $\sim$ generated by the equations in $\mathcal{E}$. By the Precise Monadicity Theorem 4.6 and lemma 4.3, it suffices to show that $\mathbf{U}$ creates coequalizers of all $\mathbf{U}$-absolute pairs. So let

$$f, g : (X, \alpha) \rightrightarrows (Y, \beta)$$

be a parallel pair of $(\mathbf{\Sigma}, \mathcal{E})$-homomorphisms such that the underlying maps $\mathbf{U}f, \mathbf{U}g$ admit an absolute coequalizer

$$\begin{tikzcd} X \arrow[r, shift left, "\mathbf{U}f"] \arrow[r, shift right, "\mathbf{U}g"'] & Y \arrow[r, "q"] & A \end{tikzcd}$$

in $\mathbf{Set}$. By absoluteness, q is preserved by every endofunctor on $\mathbf{Set}$, in particular by the signature functor $\mathbf{\Sigma} : \mathbf{Set} \to \mathbf{Set}$. Hence

$$\mathbf{\Sigma}q : \mathbf{\Sigma}Y \to \mathbf{\Sigma}A$$

is a coequalizer of the pair

$$\mathbf{\Sigma}f, \mathbf{\Sigma}g : \mathbf{\Sigma}X \rightrightarrows \mathbf{\Sigma}Y.$$

Since f and g are $(\mathbf{\Sigma}, \mathcal{E})$-homomorphisms, in particular they are $\mathbf{\Sigma}$-algebra homomorphisms, so

$$\beta \circ \mathbf{\Sigma}f = f \circ \alpha, \qquad \beta \circ \mathbf{\Sigma}g = g \circ \alpha.$$

Using that q coequalizes $\mathbf{U}f$ and $\mathbf{U}g$, we compute

$$(q \circ \beta) \circ \mathbf{\Sigma}f = q \circ \beta \circ \mathbf{\Sigma}f = q \circ f \circ \alpha = q \circ g \circ \alpha = q \circ \beta \circ \mathbf{\Sigma}g = (q \circ \beta) \circ \mathbf{\Sigma}g.$$

Thus $q \circ \beta : \mathbf{\Sigma}Y \to A$ coequalizes $\mathbf{\Sigma}f, \mathbf{\Sigma}g$, and since $\mathbf{\Sigma}q$ is their coequalizer, there exists a unique map $\gamma : \mathbf{\Sigma}A \to A$ such that

$$\gamma \circ \mathbf{\Sigma}q = q \circ \beta. \tag{5.2}$$

This endows A with the structure of a $\mathbf{\Sigma}$-algebra (A, γ), and (5.2) is precisely the condition that

$$q : (Y, \beta) \to (A, \gamma)$$

is a $\mathbf{\Sigma}$-algebra homomorphism.

To show that (A, γ) satisfies the equations $\mathcal{E}$, let $t = s$ be an equation in $\mathcal{E}$ with n free variables. For any $(\mathbf{\Sigma}, \mathcal{E})$-algebra (Z, ζ), write

$$t_Z, s_Z : Z^n \to Z$$

for the corresponding $\mathbf{\Sigma}$-term operations. It is standard (and follows by induction on the structure of terms) that for any $\mathbf{\Sigma}$-algebra homomorphism $h : (Z, \zeta) \to (Z', \zeta')$, we have

$$h \circ t_Z = t_{Z'} \circ h^n, \qquad h \circ s_Z = s_{Z'} \circ h^n.$$

Since (Y, β) is a $(\mathbf{\Sigma}, \mathcal{E})$-algebra, it satisfies $t = s$, i.e. $t_Y = s_Y$. Applying the above with $h = q : (Y, \beta) \to (A, \gamma)$, we get

$$q \circ t_Y = t_A \circ q^n, \qquad q \circ s_Y = s_A \circ q^n,$$

so

$$t_A \circ q^n = q \circ t_Y = q \circ s_Y = s_A \circ q^n.$$

Because q is an absolute coequalizer, its n-th power $q^n : Y^n \to A^n$ is again a coequalizer and hence an epimorphism in **Set**. Thus $t_A = s_A$, and the equation $t = s$ holds in (A, γ). Since $t = s$ was arbitrary, (A, γ) is a $(\boldsymbol{\Sigma}, \mathcal{E})$-algebra, which we denote $(A, \overline{\gamma})$ to stress that it satisfies all of $\mathcal{E}$.

To show the universal property of q in $(\boldsymbol{\Sigma}, \mathcal{E})$-**Alg**, let $h : (Y, \beta) \to (C, \delta)$ be a $(\boldsymbol{\Sigma}, \mathcal{E})$-homomorphism with $h \circ f = h \circ g$. Forgetting algebra structure, q is a coequalizer of $\mathbf{U}f, \mathbf{U}g$ in **Set**, so there exists a unique function $k : A \to C$ that factors h through q. Now we can see that k is a $(\boldsymbol{\Sigma}, \mathcal{E})$-homomorphism $(A, \overline{\gamma}) \to (C, \delta)$, because

$$\begin{aligned} k \circ \gamma \circ \boldsymbol{\Sigma} q &= k \circ q \circ \beta && \text{by (5.2)} \\ &= h \circ \beta && k \circ q = h \\ &= \delta \circ \boldsymbol{\Sigma} h && h \text{ is a homomorphism} \\ &= \delta \circ \boldsymbol{\Sigma}(k \circ q) = \delta \circ \boldsymbol{\Sigma} k \circ \boldsymbol{\Sigma} q. \end{aligned}$$

Because $\boldsymbol{\Sigma} q$ is a coequalizer and hence epi, we conclude

$$k \circ \gamma = \delta \circ \boldsymbol{\Sigma} k,$$

so k is a $\boldsymbol{\Sigma}$-algebra homomorphism $(A, \gamma) \to (C, \delta)$. Since (A, γ) already satisfies all equations in $\mathcal{E}$, k is in fact a $(\boldsymbol{\Sigma}, \mathcal{E})$-homomorphism.

Uniqueness of k as a homomorphism follows from uniqueness of the underlying function factoring h through q in **Set**. Hence $q : (Y, \beta) \to (A, \overline{\gamma})$ is a coequalizer of f, g in $(\boldsymbol{\Sigma}, \mathcal{E})$-**Alg**, and its underlying map $\mathbf{U}q$ is the given absolute coequalizer of $\mathbf{U}f, \mathbf{U}g$ in **Set**.

Since the choice of the **U**-absolute pair (f, g) was arbitrary, we have shown that **U** *creates* coequalizers of all **U**-absolute pairs. By lemma 4.3, **U** therefore has, preserves, and reflects coequalizers of all **U**-absolute pairs, and the hypotheses of theorem 4.6 **Ptt•** are satisfied. Thus **U** is monadic. □

In fact, the forgetful functor $\mathbf{U} : (\boldsymbol{\Sigma}, \mathcal{E})$-**Alg** $\to$ **Set** is more than just monadic, it is *strictly* monadic, as proved in [ML98].

5.2 Monads and Algebraic Theories

In the proof of theorem 5.1, we constructed the left adjoint functor $\mathbf{F} : \mathcal{A} \to \boldsymbol{\Sigma}$-**Alg** via

$$\mathbf{F}X = (\boldsymbol{\Sigma}^* X, \mu_X : \boldsymbol{\Sigma}\boldsymbol{\Sigma}^* X \to \boldsymbol{\Sigma}^* X)$$

and similarly in theorem 5.2 but with the consideration of equations. It is not hard to see that $\mathbf{\Sigma}^*$ is a monad on $\mathcal{A}$, and it is related to two concepts called *free monads* and *algebraically free monads*.

5.2.1 Free Monad

The free monad is defined in terms of a universal property, similar to free objects in general.

Definition 5.5 (Free monad)**.** Let $\mathbf{F} : \mathcal{A} \to \mathcal{A}$ be an endofunctor. A *free monad on* $\mathbf{F}$ is a monad $(\mathbf{F}^*, \eta, \mu)$ on $\mathcal{A}$ equipped with a natural transformation $\iota : \mathbf{F} \Rightarrow \mathbf{F}^*$ such that for every monad $\mathbf{T}$ on $\mathcal{A}$ and every natural transformation $\alpha : \mathbf{F} \Rightarrow \mathbf{T}$, there exists a unique monad morphism $\overline{\alpha} : \mathbf{F}^* \Rightarrow \mathbf{T}$ with $\overline{\alpha} \circ \iota = \alpha$. In particular, diagram (5.3) commutes.

$$\begin{tikzcd}
\mathbf{F} \arrow[r, Rightarrow, "\iota"] \arrow[dr, Rightarrow, "\alpha"'] & \mathbf{F}^* \arrow[d, Rightarrow, "\overline{\alpha}"] \\
& \mathbf{T}
\end{tikzcd} \tag{5.3}$$

While an algebraically free monad captures the idea of monadicity of algebras for an endofunctor.

Definition 5.6 (Algebraically free monad)**.** Let $\mathbf{T}$ be a monad on $\mathcal{A}$, $\mathbf{F}$ an endofunctor on $\mathcal{A}$. We say $\mathbf{T}$ is *algebraically free* on $\mathbf{F}$ if $\mathcal{A}^{\mathbf{T}} \simeq \mathbf{F}\text{-}\mathbf{Alg}$.

Corollary 5.1 shows that being algebraically free for a monad is a stronger condition than being free.

Corollary 5.1 (Algebraically free $\Rightarrow$ free)**.** *Let $\mathcal{A}$ be a category with finite coproducts and $\mathbf{\Sigma} : \mathcal{A} \to \mathcal{A}$ be an endofunctor, then any algebraically free monad on $\mathbf{\Sigma}$ is a free monad on $\mathbf{\Sigma}$.*

Proof. Since $\mathbf{T}$ is algebraically free on $\mathbf{\Sigma}$, by theorem 5.1 the forgetful functor $\mathbf{U} : \mathbf{\Sigma}\text{-}\mathbf{Alg} \to \mathcal{A}$ has a left adjoint $\mathbf{F} : \mathcal{A} \to \mathbf{\Sigma}\text{-}\mathbf{Alg}$ and the induced monad is $\mathbf{T} = \mathbf{UF}$. For each $X \in \mathcal{A}$, write $\Sigma^* X = \mathbf{T}X$ with structure map $\mu_X : \mathbf{\Sigma}\Sigma^* X \to \Sigma^* X$ and $(\Sigma^* X, \mu_X) = \mathbf{F}X$ is the initial $\mathbf{\Sigma}$-algebra on X.

Define a natural transformation $\iota : \mathbf{\Sigma} \Rightarrow \mathbf{T}$ by the composite

$$\begin{tikzcd}
\mathbf{\Sigma} X \arrow[r, "\mathbf{\Sigma}\eta_X"] \arrow[rr, bend right, "\iota_X"'] & \mathbf{\Sigma}\Sigma^* X \arrow[r, "\mu_X"] & \Sigma^* X
\end{tikzcd}$$

Now let $(\mathbf{M}, \eta', \mu')$ be any monad on $\mathcal{A}$ and let $\alpha : \mathbf{\Sigma} \Rightarrow \mathbf{M}$ be a natural transformation. For each object $X \in \mathcal{A}$, consider the object $\mathbf{M}X$ equipped with its canonical $\mathbf{M}$-algebra structure $\mu'_X : \mathbf{M}\mathbf{M}X \to \mathbf{M}X$. Using α, this induces a $\mathbf{\Sigma}$-algebra structure on $\mathbf{M}X$:

$$\mu'_X \circ \alpha_{\mathbf{M}X} : \mathbf{\Sigma}\mathbf{M}X \to \mathbf{M}X.$$

Since $\Sigma^* X$ is the free $\mathbf{\Sigma}$–algebra on X, the universal property gives a unique $\mathbf{\Sigma}$–algebra morphism

$$f_X : \Sigma^* X \to \mathbf{M}X$$

such that

$$f_X \circ \eta_X = \eta'_X. \tag{5.4}$$

Diagram (5.5) commutes by (5.4). It remains to show that diagram (5.6) commutes.

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[dr, "\eta'_X"'] & \Sigma^* X \arrow[d, "f_X"] \\
& \mathbf{M}X
\end{tikzcd} \tag{5.5}$$

$$\begin{tikzcd}
\Sigma^*\Sigma^* X \arrow[r, "\mu_X"] \arrow[d, "f_{\Sigma^* X}"'] & \Sigma^* X \arrow[d, "f_X"] \\
\mathbf{M}\mathbf{M}X \arrow[r, "\mu'_X"'] & \mathbf{M}X
\end{tikzcd} \tag{5.6}$$

Consider the following larger diagram:

$$\begin{tikzcd}
\mathbf{\Sigma}\Sigma^*\Sigma^* X \arrow[r, "\iota_{\Sigma^* X}"] \arrow[d, "\mathbf{\Sigma} f_{\Sigma^* X}"'] & \Sigma^*\Sigma^*\Sigma^* X \arrow[r, "\mu_{\Sigma^* X}"] \arrow[d, "f_{\Sigma^*\Sigma^* X}"] & \Sigma^*\Sigma^* X \arrow[d, "f_{\Sigma^* X}"] \\
\mathbf{\Sigma}\mathbf{M}\mathbf{M}X \arrow[r, "\alpha_{\mathbf{M}\mathbf{M}X}"] & \mathbf{M}\mathbf{M}\mathbf{M}X \arrow[r, "\mu'_{\mathbf{M}X}"] & \mathbf{M}\mathbf{M}X
\end{tikzcd} \tag{5.7}$$

The left square commutes by naturality of ι and α. The right square commutes because $f_{\Sigma^* X}$ is a $\mathbf{\Sigma}$-algebra morphism, that is,

$$f_{\Sigma^* X} \circ \mu_{\Sigma^* X} = \mu'_{\mathbf{M}X} \circ \alpha_{\mathbf{M}X} \circ \mathbf{\Sigma} f_{\Sigma^* X}.$$

Therefore the outer rectangle of (5.7) commutes. Now apply the universal property of the free $\mathbf{\Sigma}$-algebra $\Sigma^*(\Sigma^* X)$: the two composites

$$\mathbf{\Sigma}\Sigma^*\Sigma^* X \longrightarrow \mathbf{M}\mathbf{M}X$$

coincide, hence there exists a unique $\mathbf{\Sigma}$-algebra morphism

$$f_{\Sigma^* X} : \Sigma^*\Sigma^* X \to \mathbf{M}\Sigma^* X$$

making the outer rectangle commute.

Unwinding the definition of ι and of the $\mathbf{\Sigma}$-algebra structures, this is exactly the commutativity of the right-hand one in diagram (5.3). Finally, suppose $g : \mathbf{T} \Rightarrow \mathbf{M}$ is another monad morphism satisfying $g_X \circ \eta_X = \eta'_X$ for all X. Then each $g_X : \Sigma^* X \to \mathbf{M}X$ is a $\mathbf{\Sigma}$-algebra morphism agreeing with f_X on $\eta_X : X \to \Sigma^* X$. By initiality of $\Sigma^* X$, we must have $g_X = f_X$. Hence $g = f$ and f is unique. Therefore $\mathbf{T}$ satisfies the universal property of the free monad on $\mathbf{\Sigma}$. □

Corollary 5.1 in some sense proves the well-definedness of algebraically free monads, that, they are indeed free! In fact, corollary 5.1 can be generalized to any category, while the proof would need to involve higher categories [Kel80]. The other direction–that whether free monads are algebraically free–is not true in general.

5.2.2 Lawvere Theory and Algebraic Effects

Historically in the same period as the development of monads, F.W. Lawvere introduced his notion of *algebraic theories* in [Law63], also known as *Lawvere theories*, based on the observation that: algebraic theories are categories $\mathcal{T}$ with finite products whose objects are the natural numbers, and algebras are then functors from $\mathcal{T}$ to the category of sets which preserve finite products. Homomorphisms of algebras are represented by natural transformations. This provides a more generalized framework that allows higher-order reasoning about algebraic structures.

In this section, we only provide a brief introduction to Lawvere theories.

Definition 5.7 (Lawvere Theory)**.** An *Lawvere* (*algebraic*) *theory* is a small category $\mathcal{T}$ with finite products. A *model* or *algebra* for the theory $\mathcal{T}$ is a functor $\mathbf{A} : \mathcal{T} \to \mathbf{Set}$ preserving finite products. We denote by $\mathbf{Mod}\mathcal{T}$ the category of models or algebras of $\mathcal{T}$, which is a full subcategory of the functor category $[\mathcal{T}, \mathbf{Set}]$. Morphisms, also called homomorphisms, are the natural transformations.

To better understand the shift in perspective offered by Lawvere theories, we compare the classical syntactic approach to algebra with the categorical approach of functorial semantics in Table 5.1.

Concept	*Classical Algebraic View*	*Functorial Semantics (Lawvere)*
Theory	A *presentation* consisting of a signature (operation symbols) and a set of equational axioms between terms.	A *structured category* $\mathcal{T}$ with finite products. This captures the algebraic structure independent of a specific presentation.
Models	A set equipped with concrete operations that interpret the signature and satisfy the given axioms.	A functor $\mathbf{A} : \mathcal{T} \to \mathbf{Set}$ (or another category) that preserves finite products. The preservation of limits ensures axioms are satisfied.
Homomorphisms	Functions between underlying sets that respect the operations (commute with the structure).	Natural transformations between functors. This definition arises automatically from the structure of the functor category.
Logic	Relies on equational logic, consisting of substitution of equals and equivalence relation laws.	Admits *universal models* within a classifying category $\mathcal{C}_{\mathbb{T}}$, allowing for results on logical soundness and completeness relative to categorical semantics.

Table 5.1: Comparison of classical equational theories and Lawvere's functorial semantics. [AB09]

While the distinction between free and algebraically free monads addresses the internal structure of the category of algebras, a parallel development in theoretical computer science utilizes this framework to model computational semantics.

Specifically, we can view the side effects of imperative programming through the lens of algebraic operations. In imperative programming, a function $A \to B$ is often *impure* because it implicitly reads from and writes to a mutable register, called the *state*. The output depends not just on the argument A passed in, but on the current value in the state V. Furthermore, the function might change V as a *side effect*. To model this in a purely functional and categorical setting, we must make this dependency explicit, which is implemented by performing a transformation on the function signature:

> *Input*: The function requires the current state V to run. Thus, the domain becomes $V \times A$.
>
> *Output*: The function produces a result B, but also a (potentially modified) new state V. Thus, the codomain becomes $V \times B$. This transforms the impure arrow into a pure arrow:
>
> $$V \times A \longrightarrow V \times B$$

By Currying[2], we move the state input to the right side, isolating the domain A. This yields the type definition of the `State` Monad:

$$A \longrightarrow (V \to V \times B)$$

which further leads to the formal definition as follows.

Definition 5.8 (State Monad)**.** Let $\mathcal{A}$ be a Cartesian Closed Category and let V be a fixed object in $\mathcal{A}$ representing the set of possible states. The *State Monad* $\mathbf{T}$ is defined by the functor $T : \mathcal{A} \to \mathcal{A}$ where:

$$\mathbf{T}A = (V \times A)^V$$

Here, an element of $\mathbf{T}A$ is a *computation*—a function that waits for an initial state $v \in V$ and returns a pair containing the new state and the result. To see that $\mathbf{T}$ is indeed a monad, we must firstly verify that $\mathbf{T}$ is indeed an endofunctor on $\mathcal{A}$, by defining its functor action. Given a function $f : A \to B$, we need a function $\mathbf{T}f : \mathbf{T}A \to \mathbf{T}B$. This is defined by composing the internal result with f without touching the state mechanics, using the following Haskell-like pseudocode:

$$\mathbf{T}\ (fg) = \lambda v.\texttt{let}\ (v', a)\ \texttt{=}\ g(v)\ \texttt{in}\ (v', f(a))$$

[2] Often need the category to be Cartesian-closed

Then we need to show the unit η and multiplication μ. The unit transformation $\eta_A : A \to \mathbf{T}A$ represents *injecting* a pure value into the monad. In the context of `State`, this means returning a value without inspecting or modifying the state register.

$$\eta_A(a) = \lambda v.(v, a)$$

This creates a computation that leaves the state v unchanged. The multiplication $\mu_A : \mathbf{TT}A \to \mathbf{T}A$ is responsible for *flattening* layers of state. If we have $\mathbf{TT}A$, we effectively have a computation that produces another computation, which has a type of

$$V \to (V \times (V \to (V \times A)))$$

To flatten this, we run the outer layer to get the inner layer, then run the inner layer:

$$\mu_A(\varphi) = \lambda v.\mathtt{let}\ (v', \psi)\ \mathtt{=}\ \varphi(v)\ \mathtt{in}\ \psi(v')$$

This threads the state v through the first computation to get v', and then passes v' to the second computation.

The above definitions and constructions should feel natural for anyone familiar with functional programming and monads. However, monads can be harder to compose directly [JD93]. In fact, if $\mathbf{T}_1$ and $\mathbf{T}_2$ are two monads, their composition $\mathbf{T}_1 \circ \mathbf{T}_2$ is not necessarily a monad. To address this, Plotkin and Power [PP02, PP03] observed that many computational effects, including state, can be described using *algebraic operations* and *equations*. That is, (some) effects are algebras of algebraic theories, which are therefore called *effect theories*.

In particular, for the state monad, we can define two basic operations:

Definition 5.9 (Mnemoid)**.** A *mnemoid* on **Set** is an algebra with a underlying set A and the corresponding signature $\mathbf{\Sigma}$ induced by the following operations

(1) a V-ary operation

$$\mathsf{get} : A^V \to A$$

(2) a unary operation for each $v \in V$.

$$\mathsf{put}_v : A \to A$$

such that the following diagram commutes for all $v, v_1, v_2 \in V$:

$$\begin{tikzcd} A \arrow[r, "\mathsf{put}_v"] \arrow[dr, "\mathrm{id}_A"'] & A^V \arrow[d, "\mathsf{get}"] \\ & A \end{tikzcd} \tag{5.8}$$

$$\begin{tikzcd} A \arrow[r, "\mathsf{put}_{v_2}"] \arrow[dr, "\mathsf{put}_{v_2}"'] & A \arrow[d, "\mathsf{put}_{v_1}"] \\ & A \end{tikzcd} \tag{5.9}$$

$$\begin{tikzcd} A^V \arrow[r, "\mathsf{get}"] \arrow[d, "\pi_v"'] & A \arrow[d, "\mathsf{put}_v"] \\ A \arrow[r, "\mathsf{put}_v"'] & A \end{tikzcd} \tag{5.10}$$

There is an intuitive explanation[3] of these equations.

(5.8) corresponds to the equation

$$\mathsf{get}(\lambda v.\mathsf{put}_v(\kappa)) = \kappa$$

which means that if we read the state and immediately write it back, the state remains unchanged.

(5.9) corresponds to the equation

$$\mathsf{put}_{v_1}(\mathsf{put}_{v_2}(\kappa)) = \mathsf{put}_{v_2}(\kappa)$$

which means that if we write v_1 to a register and then write v_2, the first write is overwritten, and only v_2 remains.

(5.10) corresponds to the equation

$$\mathsf{put}_v(\mathsf{get}(f)) = \mathsf{put}_v(f\ v)$$

which means that if we write v to the state, and then read the state to pass it to a function f, it is equivalent to directly passing v to f and writing the result.

It turns out that these three equations are Hilbert-Post complete [Pre10], that is, if we add any equation that does not already follow from these, the theory trivializes in the sense that all equations become derivable. In 2002, Plotkin and Power [PP02] proved that the category of mnemoids is equivalent to the category of algebras of the state monad, i.e.

$$\mathbf{Mod}\mathcal{T} \simeq \mathbf{Set}^{\mathbf{T}}.$$

where $\mathcal{T}$ is the effect theory of State, and $\mathbf{T}$ is exactly the `State` monad defined in definition 5.8. The proof involves three main steps:

[3] We use continuatin-passing style because it aligns with the order of computation.

(1) establish that the state monad $\mathbf{T}$ is finitary.

(2) construct a functor

$$\iota : \Theta_{\mathbf{M}} \to \Theta_{\mathbf{T}}$$

starting from the Lawvere theory $\Theta_{\mathbf{M}}$ of mnemoids by interpreting the get and put operations and by checking that the equations are valid in $\Theta_{\mathbf{T}}$

(3) show that the functor ι is full and faithful.

This shows a deep connection between algebraic theories and monads in modeling computational effects, and provides a powerful framework for reasoning about and composing effects in programming languages.

Chapter 6

Conclusion

This thesis has explored the fundamental interplay between Free Constructions and Monadicity, uniting these concepts within the rigorous framework of Category Theory. Our objective has been to clarify how the intuitive notion of *freeness*—generating structure without unnecessary constraints—is formally realized through adjunctions and their associated monads.

A central focus of this work was the systematic analysis of Beck's Monadicity Theorem. Rather than presenting the theorem as a singular result, we explicated a hierarchy of monadicity conditions—ranging from Weak to Precise and Reflexive variants. By organizing these theorems into a coherent lattice structure, we highlighted that the *exactness* of a functor is determined by its behavior toward specific classes of coequalizers, particularly split and absolute coequalizers. This analysis demonstrates that monadicity is essentially a measure of how faithfully a functor preserves the algebraic structure inherent in these colimits.

Furthermore, we applied this machinery to Algebraic Theories, detailing the equivalence between the category of models for an equational theory and the Eilenberg-Moore category of its induced monad. We also extended this perspective to the domain of theoretical computer science, illustrating how computational effects, such as state, can be rigorously treated as algebraic structures governed by operations and equation.

Ultimately, this thesis serves to demonstrate that the abstract machinery of Monadicity is not merely a technical criterion, but the essential bridge between Syntax and Semantics. It provides the formal language to understand how algebraic structure is generated, preserved, and recognized across both mathematical and computational contexts.

Appendix A

Special Categories

Definition A.1 (Category of Sets)**.** The category **Set** has:

- *Objects*: all sets
- *Morphisms*: functions between sets
- *Composition*: composition of functions
- *Identity morphisms*: identity functions $\mathrm{id}_X : X \to X$ where $\mathrm{id}_X(x) = x$ for all $x \in X$

Proof of well-definedness. Function composition is associative: if $f : A \to B$, $g : B \to C$, and $h : C \to D$, then $(h \circ g) \circ f(x) = h(g(f(x))) = h \circ (g \circ f)(x)$ for all $x \in A$. Identity morphisms satisfy $\mathrm{id}_B \circ f = f$ and $f \circ \mathrm{id}_A = f$ for any function $f : A \to B$. □

For a family of sets $\{X_i\}_{i \in I}$, their *product* (or *cartesian product*) $\prod_{i \in I} X_i$ is the set of all functions $f : I \to \bigcup_{i \in I} X_i$ such that $f(i) \in X_i$ for all $i \in I$. For each $j \in I$, there is a canonical projection $\pi_j : \prod_{i \in I} X_i \to X_j$ defined by $\pi_j(f) = f(j)$.

Their *coproduct* (or *disjoint union*) $\coprod_{i \in I} X_i$ is the set $\bigcup_{i \in I}(X_i \times \{i\})$ with canonical inclusions $\iota_j : X_j \to \coprod_{i \in I} X_i$ defined by $\iota_j(x) = (x, j)$.

Definition A.2 (Null Object)**.** A *null object* 0 in a category $\mathcal{A}$ is an object that is both initial and terminal.

Definition A.3 (Zero Morphism)**.** Let $\mathcal{A}$ be a category with a null object 0. For any objects A, B in $\mathcal{A}$, the *zero morphism* $0 : A \to B$ is the unique morphism that factors through 0.

Definition A.4 (Kernel and Cokernel)**.** In a category $\mathcal{A}$ with a null object, the *kernel* of $f : X \to Y$ is an equalizer of f and $0 : X \to Y$. The *cokernel* of f is a coequalizer of f and 0.

Definition A.5 (Image)**.** In a category with kernels and cokernels, the *image* of f is $\operatorname{Im} f := \ker(\operatorname{coker} f)$.

Definition A.6 (Biproduct)**.** A diagram $X \xrightarrow{i_X} Z \xrightarrow{p_Y} Y$ with $p_X : Z \to X$ and $i_Y : Y \to Z$ is a *biproduct* if $p_X i_X = \mathrm{id}_X$, $i_Y p_Y = \mathrm{id}_Y$, and $p_Y i_X + p_X i_Y = \mathrm{id}_Z$.

Definition A.7 (Abelian Category)**.** A category $\mathcal{A}$ is *abelian* if: (i) it has a null object; (ii) it has binary biproducts; (iii) every morphism has a kernel and a cokernel; (iv) every monomorphism is a kernel and every epimorphism is a cokernel.

Example A.1 (Category of Abelian Groups)**.** The category $\mathbf{Ab}$ is abelian. The trivial group is a null object; direct sums give biproducts; kernels and cokernels are the usual ones; every mono is a kernel and every epi is a cokernel.

Definition A.8 (Exact and Short Exact Sequences)**.** A sequence $\cdots \to A_{n-1} \xrightarrow{f_n} A_n \xrightarrow{f_{n+1}} A_{n+1} \to \cdots$ is *exact* if $\operatorname{Im} f_n = \ker f_{n+1}$ for all n. A *short exact sequence* is $0 \to A \xrightarrow{f} B \xrightarrow{g} C \to 0$.

Definition A.9 (Split Short Exact Sequence)**.** The short exact sequence $0 \to A \xrightarrow{f} B \xrightarrow{g} C \to 0$ is *split* if there exist $s : B \to A$ and $t : C \to B$ with $sf = \mathrm{id}_A$, $gt = \mathrm{id}_C$, and $fs + tg = \mathrm{id}_B$.

Theorem A.1 (Characterization of Split Short Exact Sequences)**.** *The sequence* $0 \to A \xrightarrow{f} B \xrightarrow{g} C \to 0$ *is split iff* $B \cong A \oplus C$.

Definition A.10 (Category of Finite Sets)**.** The category $\mathsf{Set}_{\mathsf{fin}}$ has:

- *Objects*: all finite sets
- *Morphisms*: functions between finite sets
- *Composition*: composition of functions
- *Identity morphisms*: identity functions

Proof of well-definedness. This follows from the same argument as $\mathbf{Set}$, since composition of functions between finite sets remains a function between finite sets, and identity functions on finite sets are identity morphisms. □

Definition A.11 (Category of Posets)**.** The category **Pos** has:

- *Objects*: all partially ordered sets $(P, \leq)$
- *Morphisms*: order-preserving functions $f : (P, \leq_P) \to (Q, \leq_Q)$ such that $p \leq_P p'$ implies $f(p) \leq_Q f(p')$
- *Composition*: composition of functions
- *Identity morphisms*: identity functions

Proof of well-definedness. The composition of order-preserving functions is order-preserving: if $f : (P, \leq_P) \to (Q, \leq_Q)$ and $g : (Q, \leq_Q) \to (R, \leq_R)$ are order-preserving, then for $p, p' \in P$ with $p \leq_P p'$, we have $f(p) \leq_Q f(p')$ and thus $g(f(p)) \leq_R g(f(p'))$. Identity functions are trivially order-preserving. Function composition is associative by the same argument as in **Set**. □

Definition A.12 (Category of Groups)**.** The category **Grp** has:

- *Objects*: all groups
- *Morphisms*: group homomorphisms $f : G \to H$ such that $f(g_1 \cdot_G g_2) = f(g_1) \cdot_H f(g_2)$ and $f(e_G) = e_H$
- *Composition*: composition of functions
- *Identity morphisms*: identity functions

Proof of well-definedness. The composition of group homomorphisms is a group homomorphism: if $f : G \to H$ and $g : H \to K$ are homomorphisms, then $(g \circ f)(g_1 \cdot_G g_2) = g(f(g_1 \cdot_G g_2)) = g(f(g_1) \cdot_H f(g_2)) = g(f(g_1)) \cdot_K g(f(g_2)) = (g \circ f)(g_1) \cdot_K (g \circ f)(g_2)$. Identity functions are group homomorphisms. Function composition is associative as in **Set**. □

For a family of groups $\{G_i\}_{i \in I}$, their *direct product* $\prod_{i \in I} G_i$ is the group whose underlying set is the cartesian product $\prod_{i \in I} G_i$ with componentwise multiplication. For each $j \in I$, there is a canonical projection $\pi_j : \prod_{i \in I} G_i \to G_j$.

Their *free product* (coproduct in **Grp**) $*_{i \in I} G_i$ is the group of all words in the disjoint union of the G_i modulo the relations that hold within each G_i and the identification of identity elements. For each $j \in I$, there is a canonical inclusion $\iota_j : G_j \to *_{i \in I} G_i$.

Definition A.13 (Category of Topological Spaces)**.** The category **Top** has:

- *Objects*: all topological spaces (X, τ)
- *Morphisms*: continuous functions $f : (X, \tau_X) \to (Y, \tau_Y)$ such that $f^{-1}(U) \in \tau_X$ for all $U \in \tau_Y$
- *Composition*: composition of functions
- *Identity morphisms*: identity functions

Proof of well-definedness. The composition of continuous functions is continuous: if $f : (X, \tau_X) \to (Y, \tau_Y)$ and $g : (Y, \tau_Y) \to (Z, \tau_Z)$ are continuous, then for any $U \in \tau_Z$, we have $(g \circ f)^{-1}(U) = f^{-1}(g^{-1}(U)) \in \tau_X$ since $g^{-1}(U) \in \tau_Y$ and f is continuous. Identity functions are trivially continuous. Function composition is associative as in **Set**. □

For a family of topological spaces $\{(X_i, \tau_i)\}_{i \in I}$, their *product* $\prod_{i \in I} X_i$ has the product topology: the coarsest topology for which all canonical projections $\pi_j : \prod_{i \in I} X_i \to X_j$ are continuous.

Their *coproduct* (disjoint union) $\coprod_{i \in I} X_i$ has the disjoint union topology: a set $U \subseteq \coprod_{i \in I} X_i$ is open if and only if $U \cap X_i$ is open in X_i for all $i \in I$.

Definition A.14 (Category of R-Modules)**.** Let R be a ring. The category R-**Mod** has:

- *Objects*: all R-modules
- *Morphisms*: R-module homomorphisms $f : M \to N$ such that $f(m_1 + m_2) = f(m_1) + f(m_2)$ and $f(r \cdot m) = r \cdot f(m)$ for all $m, m_1, m_2 \in M$ and $r \in R$
- *Composition*: composition of functions
- *Identity morphisms*: identity functions

Proof of well-definedness. The composition of R-module homomorphisms is an R-module homomorphism: if $f : M \to N$ and $g : N \to P$ are R-module homomorphisms, then $(g \circ f)(m_1 + m_2) = g(f(m_1 + m_2)) = g(f(m_1) + f(m_2)) = g(f(m_1)) + g(f(m_2)) = (g \circ f)(m_1) + (g \circ f)(m_2)$ and $(g \circ f)(r \cdot m) = g(f(r \cdot m)) = g(r \cdot f(m)) = r \cdot g(f(m)) = r \cdot (g \circ f)(m)$. Identity functions are R-module homomorphisms. Function composition is associative as in **Set**. □

For a family of R-modules $\{M_i\}_{i \in I}$, their *direct product* $\prod_{i \in I} M_i$ is the R-module whose underlying set is the cartesian product $\prod_{i \in I} M_i$ with componentwise addition and scalar multiplication. For each $j \in I$, there is a canonical projection $\pi_j : \prod_{i \in I} M_i \to M_j$.

Their *direct sum* $\bigoplus_{i\in I} M_i$ is the R-submodule of $\prod_{i\in I} M_i$ consisting of tuples $(m_i)_{i\in I}$ where only finitely many m_i are nonzero. For each $j \in I$, there is a canonical inclusion $\iota_j : M_j \to \bigoplus_{i\in I} M_i$ sending $m \in M_j$ to the tuple with m in the j-th position and zero elsewhere.

Bibliography

[AB09] Steve Awodey and Andrej Bauer. Introduction to categorical logic. *Lecture notes*, 2009.

[AM25] Nathanael Arkor and Dylan McDermott. Relative monadicity. *Journal of Algebra*, 663:399–434, 2025.

[Awo06] Steve Awodey. *Category Theory*. Oxford University Press, Oxford, England, 2006.

[Bec03] Jonathan Mock Beck. Triples, algebras and cohomology. *Reprints in Theory and Applications of Categories*, (2):1–59, 2003. Originally published as Ph.D. thesis, Columbia University, 1967.

[BR70] Jean Bénabou and Jacques Roubaud. Monades et descente. *CR Acad. Sci. Paris Sér. AB*, 270(2), 1970.

[BW00] Michael Barr and Charles Wells. *Toposes, triples, and theories*. Springer-Verlag, 2000.

[CE99] Henri Cartan and Samuel Eilenberg. *Homological algebra*, volume 19. Princeton university press, 1999.

[Dus06] John Duskin. Variations on beck's tripleability criterion. In *Reports of the Midwest Category Seminar III*, pages 74–129. Springer, 2006.

[EM65] Samuel Eilenberg and John C Moore. Adjoint functors and triples. *Illinois Journal of Mathematics*, 9(3):381–398, 1965.

[God58] G Godement. Topologies algébrique et théorie des faisceaux. *Actualites Scientifiques et Industrielles 1252*, 1958.

[JD93] Mark P Jones and Luc Duponcheel. Composing monads. Technical report, Technical Report YALEU/DCS/RR-1004, Department of Computer Science. Yale ..., 1993.

[Kan58] Daniel M Kan. Adjoint functors. *Transactions of the American Mathematical Society*, 87(2):294–329, 1958.

[Kel80] G Max Kelly. A unified treatment of transfinite constructions for free algebras, free monoids, colimits, associated sheaves, and so on. *Bulletin of the Australian Mathematical Society*, 22(1):1–83, 1980.

[Kle65] Heinrich Kleisli. Every standard construction is induced by a pair of adjoint functors. *Proceedings of the American Mathematical Society*, 16(3):544–546, 1965.

[Law63] F William Lawvere. Functorial semantics of algebraic theories. *Proceedings of the National Academy of Sciences*, 50(5):869–872, 1963.

[Lin66] Fred EJ Linton. Some aspects of equational categories. In *Proceedings of the Conference on Categorical Algebra: La Jolla 1965*, pages 84–94. Springer, 1966.

[Mac63] Saunders MacLane. Natural associativity and commutativity. 1963.

[ML98] Saunders Mac Lane. *Categories for the working mathematician*, volume 5. Springer Science & Business Media, 1998.

[Mog88] Eugenio Moggi. *Computational lambda-calculus and monads*. University of Edinburgh, Department of Computer Science, Laboratory for ..., 1988.

[Mog91] Eugenio Moggi. Notions of computation and monads. *Inf. Comput.*, 93(1):55–92, July 1991.

[PP02] Gordon D. Plotkin and John Power. Notions of computation determine monads. In *Proceedings of the 5th International Conference on Foundations of Software Science and Computation Structures*, FoSSaCS '02, pages 342–356, Berlin, Heidelberg, 2002. Springer-Verlag.

[PP03] Gordon Plotkin and John Power. Algebraic operations and generic effects. *Applied categorical structures*, 11(1):69–94, 2003.

[Pre10] Matija Pretnar. Logic and handling of algebraic effects. 2010.

[Rie17] E. Riehl. *Category theory in context*. Aurora: Dover modern math originals. Dover Publications, 2017.

[Wad95] Philip Wadler. Monads for functional programming. In *International School on Advanced Functional Programming*, pages 24–52. Springer, 1995.