Monads in Coalgebra

Thesis submitted for the degree of

Doctor of Philosophy

at the University of Leicester

Federico De Marchi

Department of Mathematics

and Computer Science

University of Leicester

LE1 7RH

October 14, 2003

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Abstract

Universal algebra has long been regarded as a fundamental tool in studying semantics of programming languages. Within this paradigm, one can formulate statements regarding the correctness of a program by looking at the interpretations of the code in any model for the language.

While this provides a description of finite computations, other models have to be introduced in order to provide a semantics for recursion and infinite computations in general. This leads to a study of rational and infinite terms. Such terms arise by a dual construction to that of the finite ones. Namely, while the latter form an initial algebra, the former are a final coalgebra.

For this reason, it is natural to approach the study of infinite terms by dualising the categorical model of universal algebra. This leads to various different constructions, which are worth of investigation. In this thesis we approach two of them. In one case, we introduce the notions of cosignature, coequation and comodel, in the spirit of the theory of coalgebraic specification. In the second we focus on the properties of monads which can model infinitary computations. Such monads we call guarded, and include, amongst others, the monads of finite terms, infinite terms, rational terms and term graphs. As a byproduct of identifying this notion, we can solve algebraic systems of equation, which are an abstract counterpart to the notion of a recursive program scheme.

Many guarded monads we encounter are obtained by collecting, in an appropriate sense, a suitable family of coagebras. These examples are all instances of a general theorem we present, which tells under which conditions we can define a monad by a colimit operation, and when such comonads are guarded.

The level of abstraction allowed by the use of the categorical formalism allows us to instantiate some of the results in different categories, obtaining a monadic semantics for rational and infinite parallel term rewriting.

Ringraziamenti

Se sono arrivato fin qui, e ci sono arrivato in questo modo, lo devo innanzitutto ai miei genitori, a mia nonna, ed ai piccoli e grandi sacrifici che hanno fatto per me. Ringraziarli e' davvero il minimo che possa fare, ed è in qualche modo inadeguato alla riconoscenza che provo.

Poi c'è la mia formazione "professionale", per cui devo ringraziare la professoressa Ornella Sarocchi, che ha sempre appoggiato la mia vena matematica, tutti i docenti dell'Università di Genova e in particolare i Professori Giuseppe Rosolini e Marco Grandis, miei mentori in teoria delle categorie.

L'esperienza inglese, però, è stata per me molto più che un'occasione professionale. Le persone che dall'Italia o da altrove, in momenti diversi, mi sono state vicine sono troppe per citarle tutte, a cominciare dagli zii Gianna, Alfredo e Claudia, e dai cugini Lara, Andrea, Max e Rinaldo, per continuare con la schiera innumerevole di amici. Un grazie sentito va a Steppa ed Emilio, per la fraterna amicizia e presenza dimostrata in tutti i momenti più delicati. A Matteo, Orietta e tutta la banda dei ravatti. Ad Elisa, per i suoi sinceri consigli. A Giulio, Daniele, Paola, Irene, Silvia, Veronica, e tutti gli altri dottorandi del gruppo. A Fiorina e Federico: la coppia più bella del mondo. Ad Erika, Titty e Andrea, Giovanna, Nicola, Cecco, Myriam, Piola, Valeria, Massimo, Alessandra, Giorgio, Francesco, Elvira e tutti gli altri che sto dimenticando ma che ci sono sempre stati per ogni mio ritorno. A tutti voi un sincero e sentito ringraziamento, perché senza di voi non ce l'avrei mai fatta.

Leicester, Giugno 2003

Aknowledgements

So, after three years, time has come for me to leave Leicester, the rain, the lager, the smell of curry... well, I suppose it had to happen, at some point! It has been for me a great time, and of this I have to thank first of all my supervisor Neil Ghani, for having introduced me to the mystery of beer at five and having lead my first steps through computer science.

Then all those in the Department who have supported my research more or less directly. Amongst the others, Fer-Jan de Vries, Simon Ambler, Roy Crole, Vincent Schmitt, and our Head of Department Prof Rick Thomas. For having sustained my studies, I am of course thankful to EPSRC.

For that little English I have learnt, I will never thank enough my coursemate Gabriel Davis. With him, I hereby thank also all the other PhD students of the department, whom I walked my path with. Then there are friends. I'll start with the Italians: Raffa (more than a flat-mate: a full "monella" sister!), Catax, Marcello, Federica and Alessandra (to name a few). Then the foreigners: Sam (my squasher!), Gemma, Anna (my Spanish teacher and much more!), Dr Frank-supercar-Neumann, Kristina, Noelia, Nuria, Nadia, Fei (a far but close friend), Raul, Jean, and all those who I am forgetting now.

Thank you all, for having made these three years so special and unforgettable, and for having been such good and supportive friends to me.

Leicester, June 2003

Contents

Chapter 1

Introduction and Preliminaries

In the continuous process of abstraction which moved mathematics since its very origins, it has been a common practice to identify relations between different entities and the equalities which these entities satisfy. It soon became clear that some relations determine the dependency of some elements on others. The concept of function was invented to formalise this idea, but for long time they have only been relating numbers. Only relatively recently Galois realised that some properties of particular functions (namely, the sum) were not dependent on the numbers themselves; he first identified the structure of a group. Roughly at the same time, the notion of a vector space was invented, and within a century, several others appeared, in order to model and reason about properties of mathematical objects of practical interest. Among them, monoids, rings, modules, and the like.

Half a century later, people started realising that these were all instances of a more general theory, independent of the specific functions and equations. The theory of universal algebra took shape from these premises, around the beginning of the twentieth century. At the earliest age of computing, people understood that in principle there was no real difference between considering operations on sets and key-words in a programming language acting on a configuration of memories in a machine. Soon, they employed the techniques of universal algebra to give a semantics to computation. Under this impulse, the study of universal algebra got a significant thrust.

With the introduction of the categorical formalism, two more abstractions became possible. On the one hand, by using monads to model an algebraic theory we can capture precisely the key properties of the notion of substitution. On the other hand, the achieved generality allows instantiations of the theory on entities other than sets. Amongst the examples of structures which can be modelled within this framework, we have multi-sorted theories, categories with structures, Boolean algebras and other families of partial orders, term rewriting systems, variable binding [24, 43, 51, 20].

One of the main logical contributions of universal algebra has definitely been the exhaustive explanation of the induction principle. However, in the mathematical practice, as well as in computer science, one often needs other proof principles in order to prove the desired results. Often, things go wrong when considering infinitary structures, which can not be obtained by a finite iteration of the operations. These structures, though, are essential to the theory of computing, both in modelling infinite data (such as lists or streams) and in reasoning about the behaviour of programs whose computation need not terminate. Amongst the proof principles used in reasoning with these objects, there are Banach's fixpoint theorem and the notion of observational equivalence. These notions are somehow dual to the induction principle, in much the same way as the infinitary structures which they consider are dual to the finite terms. The use of category theory makes this duality explicit, for if universal algebra is usually associated to the theory of algebras, infinite terms are associated with coalgebras.

The purpose of this thesis is that of exploring how the monadic approach to universal algebra can be dualised in order to model and reason about infinitary structures, and how much of the expressivity of the monadic model can be transposed through this process. Given that the monad associated to an algebraic theory is pointwise calculated as an initial algebra, it is natural to focus our attention on the categorical dual of this notion; that is, on final coalgebras.

The Categorical Approach to Universal Algebra

The classic categorical approach to universal algebra considers, for a given signature Σ, a finitary endofunctor F_Σ, which associates with each set of variables the set of ground terms built over them. Algebras for the functor F_Σ are models for the theory generated by the signature, and given variables from a set X, the set $T_\Sigma X$ of finite Σ-terms over X plays a key role, being the free F_Σ-algebra on X. Its universal property captures precisely the essence of the induction principle, and by these means one can prove that the association $X \mapsto T_\Sigma X$ defines a monad. Algebras for the monad are again the same as models for the signature, but their extra structure allows us to talk about the algebraic properties of such models in a much cleaner way. The monad arising is the free one over F_Σ, and provides a semantics for the signature.

Adding equations to a theory is equivalent to consider another signature E and the free monad induced by it. Roughly speaking, this monad will build the proof-terms we need in order to show that two terms are equal in the algebraic theory determined by $\langle \Sigma, E \rangle$. The left and right handsides of the equations are determined by two monad morphisms from T_E to T_Σ, and their coequaliser in the category of monads has as algebras precisely the models of $\langle \Sigma, E \rangle$. This explains how monads (in fact, finitary monads) model categorically the concept of an algebraic theory. At this stage, category theory is used mainly as a formalism. Its precise way of fitting the set-theoretic notions, and the immediacy of its language make the categorical approach very useful in encoding and proving the universal properties of the algebras, but add no more expressivity.

A substantial improvement is due to the work by Kelly and Power. In a paper which can be considered as the cornerstone of the present work [37], they show how the categorical transpositions of the notions of signature, equation and model make sense not just for the category of sets, but in general for any enriched category with enough structure. On top of this, they prove that any monad on such categories does in fact model some algebraic theory.

In a good review of that paper by Edmund Robinson [51], several applications are proposed, and this shows the gained expressivity, since we can now identify as models of algebraic theories such structures as categories with structure, ω-CPO's, term rewriting systems [43], and many others.

Cofree Comonads

The theory presented by Kelly and Power consists of several steps, each of which potentially allowing a dualisation. In this work, we choose to concentrate on the one leading to the consideration of F_Σ-algebras, and replace them by coalgebras, exploring what structures we can derive from them and what they represent from a computational perspective. The reason for choosing coalgebras is that, as clearly shown by the research in this field over the last decade [57, 31], they capture the idea of infinite computation and infinite data. Their duality with respect to algebras is that of infinite versus finite.

However, starting from F_Σ there are two steps leading to the free monad T_Σ over it. One is to consider F_Σ-algebras; the second is to consider the collection of the free ones. In fact, the free algebra on an object X is also the initial $X + F_\Sigma$-algebra. Having considered coalgebras instead of algebras, we now face the choice of dualising the next step in two different ways. One is to consider the cofree coalgebras (i.e. the final $X \times F_\Sigma$-coalgebras), the other is to consider the final $X + F_\Sigma$-coalgebra.

In this thesis we analyse in detail the structure generated by any of these choices. In the first case, the collection of cofree coalgebras defines the cofree comonad on F_Σ, and much of the picture for the algebraic case can be recovered. The coequaliser which we considered before now becomes an equaliser, and its syntactic counterpart is that of a coequation.

The kind of structures captured by this framework are in the area of coalgebraic specification, where people define cosignatures and coequations, and then look at the behaviours of processes defined according to their specification [17, 11].

Unfortunately, because of the leap in the size of our structures when moving from algebras to coalgebras, we can no longer prove that every monad models a theory.

Monads of Infinite Terms

If we choose the other possible dualisation, we find out that the functor taking an object X to the final $X + F_\Sigma$-coalgebra carries the structure of a monad, and a nice one too. Its action on X builds the set of finite and infinite Σ-terms over X. The advantage of this approach is that, once more, the monadic structure precisely captures the key properties of substitution of terms.

Algebras for this monad are models of the signature Σ where we can interpret also infinite iterations of function symbols as functions. The relevance of this structure [25, 2], is that it allows us to find a unique solution to recursive equations: a property which is desirable in computer science, where many programs are defined by means of recursive programs schemes. Moreover, being pointwise a final coalgebra, the monad arising here provides some kind of semantics for behaviours, in that the evolution of any system defined by Σ-actions determines a unique infinite term, which can be thought of as its semantics.

Monads of Coalgebras

The collection of all finite and infinite terms is not the only syntactic structure which can be described by means of coalgebras. In fact, despite their universal property and the fact that they offer a semantics for all processes described by Σ, infinite terms are not convenient in the computing practice, because they can not be given a finite description. It is therefore reasonable to investigate other structures which can be described by a finite amount of data. Rational terms, for instance, are modelled by coalgebras with a finite carrier set. The image of the unique morphism from such a coalgebra to the final one determines precisely the term which it describes. By taking the collection of all such coalgebras (in a suitable categorical sense), we shall determine a new monad, which lies in between the free one and the one of infinite terms.

More generally, in this thesis we shall introduce the notions of guarded and strongly guarded monads, to identify and study properties of those monads which can be thought of as monads of terms for some signature, and we shall give enough conditions for a collection of coalgebras to determine a monad, and for such a monad to be guarded or strongly guarded.

As another instance of this, we will be able to obtain the monad of term graphs, where the collection of coalgebras is now chosen in such a way that we can model the idea of sharing the resources, by allowing parallel edges between nodes, and that of recursion, by loops.

Amongst the results, we show that for strongly guarded monads we can solve any algebraic system of equations, i.e. we can define functions by means of recursive program schemes.

Equations and Rewriting

In general, imposing equations on infinite terms is not as easy as with the finite ones. That is because infinite terms arise as limits of sequences of finite terms, but the equivalence relations are in general not transfinite, therefore they cannot be extended to the limit. Therefore, much of the expressivity of the Kelly-Power framework fails to extend when considering other than free monads.

However, in the category of preorders we can give a nice and intuitive interpretation of the coequaliser of two monads of infinite terms or of rational terms. Ghani and Lüth [43, 42] showed how a term rewriting system can be modelled by a coequaliser of free monads on the category of preorders. We show here how, by replacing the free monads with the corresponding ones for rational terms or infinite terms, we can model the classical notions of rational term rewriting and of infinite parallel term rewriting, gaining some levels of generality and certainly a good deal of clarity with respect to the standard definitions.

Synopsis

The thesis is organised as follows. In the remainder of this Chapter we shall first briefly recall the main set-theoretical concepts in universal algebra and the theory of rewriting; then we shall revisit those notions in a categorical perspective, introducing along the way all the notions which will be employed throughout the dissertation. The exposition is intended to reach as a wide audience as possible, and in particular those who are less familiar to the language of category theory. For this reason, we shall assume knowledge only of the very basic notions of category, functor, natural transformation, adjunction, limit and colimit as they can be found in [44], and build thereupon a categorical semantic for universal algebra, going through the paper by Kelly and Power mentioned above.

In Chapter 2 we shall explore the two different dualisations of that theory we described above.

Following on the study of the monad of finite and infinite terms, in Chapter 3 we shall introduce the notion of F-guarded and strongly F-guarded monad, and we show examples. The second half of the Chapter is devoted to the proof of a theorem giving enough conditions to define a monad as a pointwise colimit.

As examples of applications of the theorem we present the monad of rational terms and that of term graphs, deriving also their guardedness properties. This is the content of Chapter 4.

Finally, in Chapter 5 we extend the Kelly-Power framework to other than free monads, and show how this can model different forms of parallel rewriting.

Some parts of this work have been published previously in joint works with my supervisor Neil Ghani, Christoph Lüth and John Power [45, 26, 25]. That material has been rearranged here, and some parts were reworked. The results in the last Chapter are entirely new, to our knowledge.

1.1 Classical Notions

The purpose of this section is to introduce and fix the notations for the classical notions of signature, equation and model for algebraic theories. We shall also define the main families of terms which we shall encounter in the rest of the thesis, i.e. finite, infinite and rational terms, and give some of their properties. Most of these notions can be found in any book on universal algebra. We refer the interested reader to the comprehensive presentations written by Courcelle [19] and Elgot [21]. At the end of the section we shall also give the definition of term rewriting systems and their models. A good reference for this subject is [39].

1.1.1 Universal Algebra

A *signature* Σ consists of a set of *function symbols*, each with a specified *arity*. The arity specifies the number of arguments which the symbol is meant to take when interpreted as a function. In most of our examples, this will be a finite number, and for this reason we shall sometimes say that the signature is *finitary*. An alternative way of presenting a signature is as a "map" associating with each arity the set of function symbols with that arity. We shall sometimes indicate by Σ_n the set of elements in Σ which have arity n (those terms we also call n-ary). A symbol of arity 0 is called a *constant*.

Given a signature Σ and a set of *variables* X, we can inductively define the set $T_\Sigma X$ of *finite terms* over Σ with variables in X as follows:

$$\frac{x \in X}{x \in T_\Sigma X} \qquad\qquad \frac{f \in \Sigma_n \quad t_1, \ldots, t_n \in T_\Sigma X}{f(t_1, \ldots, t_n) \in T_\Sigma X} \tag{1.1}$$

A term is called *ground* if it is obtained by a single instance of the right rule above on a function symbol f and variables $x_1, \ldots, x_n$. It is called *closed* if it contains no variables.

An *equation* $X \vdash t = s$ consists of a finite set of variables X (identifying the *context* where the equation holds) and a pair of terms t and s with variables from X. A *(finitary) algebraic theory* consists of a pair $\langle \Sigma, E \rangle$ where Σ is a (finitary) signature and E is a finite set of equations between terms over Σ.

Example 1.1 Most of the algebraic structures in mathematics are examples of algebraic theories. For example, groups are defined by a binary operation $*$, a unary operation ${}^{-1}$ and a constant e, subject to the following equations:

$$\text{ASS: } \{x, y, z\} \vdash x * (y * z) = (x * y) * z$$

$$\text{LUN: } \{x\} \vdash x = e * x \qquad \text{RUN: } \{x\} \vdash x = x * e$$

$$\text{LINV: } \{x\} \vdash x * (x)^{-1} = e \qquad \text{RINV: } \{x\} \vdash (x)^{-1} * x = e.$$

When we prove properties of an algebraic theory only by means of the equations and the term constructors defining it, we are using its equational logic; however, in some cases it is easier to give a proof by reasoning on the models of the theory itself.

A *model* of a signature is a set A together with an *interpretation* of any n-ary function symbol f as a function $[\![f]\!]: A^n \longrightarrow A$. A *morphism of models* is a map between sets which respects the interpretation of the symbols; i.e. if $\phi: (A, [\![\ ]\!]_A) \longrightarrow B, [\![\ ]\!]_B)$ is a morphism of models, then for any n-ary symbol f and elements $a_1, \ldots, a_n \in A$ we have $\phi([\![f]\!]_A(a_1, \ldots, a_n)) = [\![f]\!]_B(\phi(a_1), \ldots, \phi(a_n))$.

For any signature Σ, the set $T_\Sigma X$ is clearly a model, where the interpretation of $f \in \Sigma_n$ is defined by $[\![f]\!](t_1,\ldots,t_n) = f(t_1,\ldots,t_n)$ for $t_i \in T_\Sigma X$. Given a function $\nu\colon X \longrightarrow A$, from a set of variables X to a model A, this inductively defines a map $[\![\ ]\!]_\nu$ from $T_\Sigma X$ to A as follows:

$$\frac{x \in X}{[\![x]\!]_\nu = \nu(x)} \qquad \frac{f \in \Sigma_n \qquad t_1,\ldots,t_n \in T_\Sigma X}{[\![f(t_1,\ldots,t_n)]\!]_\nu = [\![f]\!]([\![t_1]\!]_\nu,\ldots,[\![t_n]\!]_\nu)}$$

It is a trivial observation that the induced map $[\![\ ]\!]_\nu$ is a model morphism and that it is the only one extending the function ν.

We call *substitution* a function $\sigma\colon X \longrightarrow T_\Sigma Y$ mapping each variable in X to some term with variables from Y. Such a σ determines a map (which we indicate in the same way) between $T_\Sigma X$ and $T_\Sigma Y$. We say that $\sigma(t)$ is the application of the substitution σ to $t \in T_\Sigma X$. Sintactically, this is defined by the following clauses:

$$\frac{t = x \in X}{\sigma(t) = \sigma(x)} \qquad \frac{f \in \Sigma_n \qquad t_1,\ldots,t_n \in T_\Sigma X}{\sigma(f(t_1,\ldots,t_n)) = f(\sigma(t_1),\ldots,\sigma(t_n))}$$

A model A of Σ *satisfies* an equation $X \vdash t = s$ if both sides of it are interpreted by the same element of A, i.e. if for any $\nu\colon X \longrightarrow A$ we have $[\![t]\!]_\nu = [\![s]\!]_\nu$. A *model* of an algebraic theory $\langle \Sigma, E\rangle$ consists of a set M together with an interpretation of the function symbols which satisfies all the equations in E.

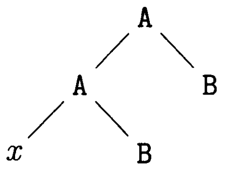
In presence of a set of equations E, we can consider the equivalence relation $\sim$ on $T_\Sigma X$ defined by the following rules:

$$\frac{t_i \sim s_i \quad (i = 1,\ldots,n) \quad f \in \Sigma_n}{f(t_1,\ldots,t_n) \sim f(s_1,\ldots,s_n)} \qquad \frac{Y \vdash t = s \qquad \sigma\colon Y \longrightarrow T_\Sigma X}{\sigma(t) \sim \sigma(s)}$$

The maps interpreting function symbols in $T_\Sigma X$ then induce maps on the quotient $T_\Sigma X/\!\sim$, and these form a model of the algebraic theory $\langle \Sigma, E \rangle$.

Models and their morphisms define the *category of models* for the algebraic theory $\langle \Sigma, E \rangle$. The connection between the equational logic of a theory and its models is that two terms t and s in $T_\Sigma X$ are provably equal by means of the equations if and only if for any model A of the theory and for any map $\nu : X \longrightarrow A$ we have $[\![t]\!]_\nu = [\![s]\!]_\nu$.

We shall often refer to terms by their equivalent representation as *syntax trees*. A tree is given by a set of *nodes* labelled by symbols in Σ or variables, and an ordered set of *branches*, so that from any node there are exactly as many branches departing as the arity of the function symbol labelling it. Nodes labelled by constants (or by variables) are called *leaves*. Pictorially, we shall identify the nodes by their labels and the branches by some edges. The most relevant difference between a tree and a graph is that a tree is rooted, connected, and does not have loops or parallel edges. For example, if $\mathtt{A}$ is a binary symbol and $\mathtt{B}$ is a constant, then the term $\mathtt{A}(\mathtt{A}(x, \mathtt{B}), \mathtt{B})$ on the variable x is depicted as the tree



A tree is *finite* if so is the set of its nodes. Otherwise, it is *infinite*. The *subtree* of a tree t at the node N is the tree whose nodes are all the descendants of N (i.e. those nodes which occur after a finite number of branches starting from N) labelled by the same symbols and joined by the same branches. A tree is called *rational* if it has only a finite number of subtrees (therefore all finite trees are rational).

1.1.2 Metric, Ordering and Completions of Terms

We shall henceforth deliberately use the words tree and term without any distinction.

The tree notation makes it easier to introduce the concept of *depth* for terms. We say that a particular occurrence of a function symbol or variable in a term t is at depth n if it occurs after n branches starting from the root of t. So, for example, the variable x in the term above is at depth 2. The root of a term is always at depth 0. The depth of a finite term is the highest depth of its leaves.

Using depth, we can define a metric on the set of finite terms, by saying that the distance between the terms t and u is 2^{-n}, where n is the least depth at which a node of t is labelled differently from the corresponding node in u. So, for example, the distance between $\mathtt{A}(x, y)$ and $\mathtt{A}(x, x)$ is $1/2$.

It is not difficult to show that a Cauchy sequence of terms, within this metric is either definitely stable or consists of terms whose states of depth up to n are fixed from some point onwards, for each natural number n. Therefore, we can identify the limit of such a sequence with the *infinite term* modelled by the infinite tree whose nodes are the ones which get fixed along the sequence. For example, the sequence $(\mathtt{A}^n\mathtt{B})_{n\in\mathrm{N}}$ converges to the infinite term $\mathtt{A}^\omega = \mathtt{A}(\mathtt{A}(\mathtt{A}(\ldots)))$. In this way we can define the set $T_\Sigma^\nu X$ of finite and infinite terms and prove the following [10]:

Proposition 1.2 *The set $T_\Sigma^\nu X$ of finite and infinite terms over X is the*

Cauchy completion of $T_\Sigma X$ with the metric defined above.

Another form of completion which $T^\nu_\Sigma X$ enjoys with respect to $T_\Sigma X$ is related to an order on terms. Given a signature Σ, we introduce a fresh constant symbol $\perp$ (called *bottom*) and form the set of terms over $\Sigma \cup \{\perp\}$ with variables in X. On these we can define the operation of *truncation* for a term at depth n. This is achieved by relabelling with $\perp$ all those nodes at depth n which are not labelled by constants and deleting all the descendants of those nodes. We define a partial order $\preceq$ on terms by saying that $t \preceq u$ if t is the result of truncating u at depth n for some $n \in \mathbb{N}$. Analogously to the metric case, one can prove that infinite terms form the ideal completion of the set of finite terms over this enhanced signature [29]. Note that $\perp$ is the least element of the partial order.

1.1.3 Recursive Systems of Equations

A term t is called *guarded* if its root is labelled by a function symbol, i.e. if it is not a variable.

A *system of equations* is *guarded* or *ideal* or in *Greibach normal form* if it has the form

$$(x_i \approx t_i)_{i \in I} \tag{1.2}$$

where $X = \{x_i \mid i \in I\}$ is the set of *unknowns* for the system, and the t_i's are guarded terms with variables in $X \cup Y$, for a set of *parameters* Y disjoint from X. A system is *finite* whenever I is. A *solution* to such a system is a substitution $\sigma: X \longrightarrow T^\nu_\Sigma Y$ such that, for each $i \in I$, $\sigma(x_i) = \sigma(t_i)$.

The possibility of solving guarded systems of equations is fundamental when studying the semantics of programming languages, because they provide an essential tool to model recursion [10, 22, 15]. The study of recursion lead several researchers to the definition of *iterative* and *iteration* theories, where solutions to such systems are guaranteed to exist. Since in our work we will only marginally touch the subject, we give here a very particular definition of an iterative theory. For us, an *iterative theory* $T \subset T^{\nu}_{\Sigma}Y$ is a set of terms with variables in Y, which is closed under substitution, contains the variables, and it is such that any finite guarded system of equations where the right handsides are in T admits a unique solution in T itself. Such a T is a *completely iterative theory* if existence and uniqueness of solutions is ensured for *all* guarded systems (i.e. not just for the finite ones).

A finite guarded system where the t_i's are infinite terms can easily be seen as an endofunction on the Cauchy complete metric space $T^{\nu}_{\Sigma}(X \cup Y)^I$, and guardedness makes this function contractive. Therefore, by Banach's theorem, it admits a unique fixpoint, and that is the solution of the system. An extension of this result to the case where I is infinite shows that the set of finite and infinite terms is the smallest set of Σ-terms closed under solution of guarded systems; i.e. any system where the t_i's are finite or infinite trees has a solution in $T^{\nu}_{\Sigma}Y$ and this is the smallest set with this property [21].

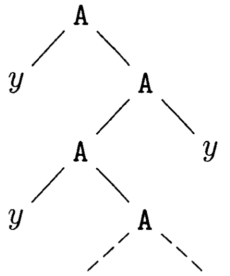
If we restrict our attention to finite systems, then the smallest set of terms which is closed under solution of guarded systems is the set $R_{\Sigma}Y$ of rational terms with variables in Y as introduced before [21, 28].

For example, the system

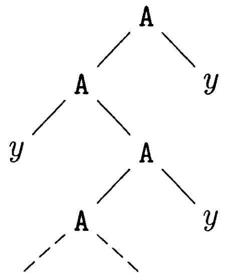
$$\begin{aligned} x_1 &\approx \mathtt{A}(y, x_2) \\ x_2 &\approx \mathtt{A}(x_1, y) \end{aligned} \tag{1.3}$$

has as a solution the rational terms

$x_1 =$



$x_2 =$



1.1.4 Term Rewriting Systems

The last notion we want to introduce, before proceeding towards a categorical reformulation of the theory is that of term rewriting systems.

The idea of term rewriting is to model computation by giving a preorder relation on the set of terms, where a term precedes another if the first computes (in one or many steps) to the second.

Formally, a *term rewriting system* (TRS) is given by a pair $\langle \Sigma, \mathcal{R} \rangle$, where Σ is a finite signature and $\mathcal{R}$ is a set of *rules* of the form $X \vdash \rho \colon t \dashrightarrow s$, where X is a set, ρ is the name of the rule, and t and s are Σ-terms with variables in X.

Given a TRS $\langle \Sigma, \mathcal{R} \rangle$, we can inductively define the *one-step reduction relation* $\dashrightarrow$ on the set $T_\Sigma X$ of finite terms over X by the following clauses:

$$\frac{\sigma \colon X_i \longrightarrow T_\Sigma X}{\sigma(t_i) \dashrightarrow \sigma(s_i)} X_i \vdash \rho_i \colon t_i \dashrightarrow s_i \in \mathcal{R}$$

$$\frac{t \dashrightarrow s \quad f \in \Sigma_n}{f(t_1, \ldots, t, \ldots, t_n) \dashrightarrow f(s_1, \ldots, s, \ldots, s_n)} \tag{1.4}$$

The *many-steps reduction relation* is the reflexive and transitive closure $\rightarrow^*$ of $\rightarrow$, and determines a preorder on $T_\Sigma X$, which models the computation of any Σ-term (seen as a program).

1.2 The Categorical Model of Universal Algebra

As we mentioned already, we shall skip the basic definitions in the field, and refer the reader to Mac Lane's book [44] for further details on the notions of category, functor, natural transformation, limit, colimit, and adjunction, as well as the main results concerning them.

To fix notation, we recall that a diagram in a category C with a collection of objects $|\mathsf{C}|$ is given by a functor X from a category D of indexes to C. We shall denote the image of such a functor on an object d in D by X_d, and for a map $k: d \longrightarrow d'$ in D we shall get a map $X_k: X_d \longrightarrow X_{d'}$ in C. If Y is a colimit of the diagram, then we write the d-th component of the colimiting cocone as the map $\overline{d}: X_d \longrightarrow Y$.

Our purpose in this section is to explain the framework of Kelly and Power [37] and explain how the syntactic notions which we outlined in the previous section are indeed captured by this framework. In order to achieve this goal, we shall have to introduce several categorical structures together with some results concerning them. Some of them are already present in Mac Lane's book, but we shall sometimes recall them here in order to fix notation, since they will be essential throughout the thesis.

1.2.1 Locally Presentable Categories

The first notion we have to render categorically, is that of a signature, and even before that, of an arity. Locally presentable categories provide the appropriate structure to formalise such concepts. The easiest way to introduce the notion, though, is probably by thinking of them as generalised algebraic lattices. The definitions we give in this section are taken from Adámek and Rosicky's book [8], to which we refer for details of the proofs and for further developments.

A cardinal λ is *regular* if every set of the form $\bigcup_{i \in \mathcal{I}} X_i$ has cardinality less than λ whenever $\mathcal{I}$ and each X_i have cardinality less than λ. Regular cardinals provide a bound for size inside a diagram. In particular, we are interested in those diagrams whose colimit is obtained by successive approximations of subdiagrams of size a regular cardinal.

Remark 1.3 From now on, unless explicitly stated, a cardinal λ will always be assumed to be regular.

Definition 1.4 A category I is *λ-filtered* if for any set J of less than λ objects in I there is an object i in I and maps $f_j\colon j \longrightarrow i$ for any $j \in J$ and for any set L of less than λ parallel arrows $f_l\colon i \longrightarrow j$ in I there is a map $g\colon j \longrightarrow h$ in I such that the composites gf_l are all equal. A *λ-filtered* diagram in a category C is a diagram $C\colon \mathsf{I} \longrightarrow \mathsf{C}$ with I λ-filtered. A colimit of a λ-filtered diagram is said to be λ-filtered. An ω-filtered diagram is referred to just as a *filtered diagram.*

Example 1.5 We present a couple of examples, in order to show what a filtered colimit is in Set and what the notion reduces to in the case of a preorder.

1. Suppose $F\colon D \longrightarrow \mathsf{Set}$ is a filtered diagram. Then its colimit is given by the disjoint union of F_d $(d \in D)$, quotiented by the transitive closure of the relation which says that, for $x \in F_d$ and $x' \in F_{d'}$, $x \sim x'$ if and only if there exist $d'' \in D$ and arrows $f\colon d \longrightarrow d''$, $f'\colon d' \longrightarrow d''$ such that $Ff(x) = Ff'(x')$.
   Using this, one can easily show that each set is the filtered colimit of its finite subsets ordered by inclusion.

2. In a partial order X considered as a category, a diagram is filtered if and only if, any pair of elements has an upper bound, i.e. if it is a directed subset of X. A filtered colimit is then the least upper bound of a directed subset.

Functors preserving λ-filtered colimits play an important role, and they deserve a special name.

Definition 1.6 A functor is *λ-accessible* if and only if it preserves λ-filtered colimits. A functor is *accessible* if it is λ-accessible for some λ. The least λ for which a functor is λ-accessible is called its *rank*. ω-accessible endofunctors are often called *finitary*.

Given a cocomplete category, it is easy to identify a particular class of objects, which will then play the role of our arities. The intuition, here, is again order theoretic. There, an element of a partial order is called *finite* if, whenever it is smaller than the least upper bound of a directed set, there is an element in the subset which is above the element itself. Generalising this to a categorical setting, we get the following.

Definition 1.7 An object K of a category C is called *λ-presentable* if its hom-functor $\hom(K,-)\colon \mathsf{C} \longrightarrow \mathsf{Set}$ is λ-accessible. If $\hom(K,-)$ is finitary, then K is said to be *finitely presentable.*

Remark 1.8 Explicitly, preservation of filtered colimits for a hom-functor $\mathsf{C}(K,-)$ means that for each filtered colimit $(D_i \xrightarrow{d_i} C)_{i\in I}$ and each morphism $f\colon K \longrightarrow C$ there exists $i \in I$ such that, as can easily be shown by the characterisation given in Example 1.5,

1. f factors through d_i, i.e. there exists $g\colon K \longrightarrow D_i$ such that $d_i g = f$;
2. the factorisation is essentially unique, i.e. if $f = d_i g$, $f = d_i g'$ then there exists $j \in I$, $d\colon i \longrightarrow j$ such that $D(d)g = D(d)g'$.

Example 1.9 We extend the two examples above and explain what finitely presentable objects are in those cases.

1. In Set, finitely presentable objects are precisely finite sets. Since each set is the filtered colimit of the inclusion-ordered set of its finite subsets (see Example 1.5), considered as a category, if a set K is finitely presentable, then the identity map $\mathsf{id}_K\colon K \longrightarrow K$ must factor through a finite subset S of K, hence K must be finite.

2. In the case of complete partial orders, finitely presentable objects turn out to be the *finite* elements (see for example [32], for details).

We now have all the elements to define what a locally presentable category is. The idea is that the category is determined by its λ-presentable objects, by means of colimits.

Definition 1.10 A category C is *locally λ-presentable* (lλp) if:

1. C is cocomplete;
2. every object in C is a λ-filtered colimit of λ-presentable objects;
3. λ-presentable objects form, up to isomorphism, only a set.

When $\lambda = \omega$, we say that C is *locally finitely presentable* (lfp). A category is said to be *locally presentable* if it is locally λ-presentable for some λ.

Example 1.11 **Set** is locally finitely presentable, and this trivially follows from Example 1.5 and Example 1.9. A partial order (considered as a category) is lfp if and only if it is an *algebraic lattice*, i.e. every element is the directed supremum (filtered colimit) of the set of finite (finitely presentable) elements lower than it.

The following result states some important properties of locally presentable categories. Its proof can be found in [8].

Theorem 1.12 *Locally presentable categories are complete, as well as cocomplete, well-powered (i.e. each object has – up to isomorphism – only a set of subobjects), and well-copowered (the dual notion). Moreover, every object in a locally presentable category is λ-presentable for some λ.*

A further relaxation of the notion, which will prove fruitful in the next chapters, is that of *accessible categories*, where cocompleteness is dropped, and only filtered colimits are required to exist.

Definition 1.13 A category C is λ-*accessible* provided it has λ-filtered colimits and a set C_λ of λ-presentable colimits such that each object in C is a λ-filtered colimit of objects in C_λ.

A category is *accessible* if it is λ-accessible for some λ.

Proposition 1.14 *A category is locally presentable if and only if it is accessible and cocomplete.*

For a given lλp category C, we shall henceforth write $\mathcal{N}_\lambda$ for the discrete category whose objects are representatives, up to isomorphism, of λ-presentable objects (and we shall write $\mathcal{N}$ when $\lambda = \omega$). The full subcategory of C on λ-presentable objects will be denoted C_λ (and C_{fp} when $\lambda = \omega$). The inclusion functors relating them are denoted $I_\lambda \colon \mathcal{N}_\lambda \longrightarrow \mathsf{C}_\lambda$ and $J_\lambda \colon \mathsf{C}_\lambda \longrightarrow \mathsf{C}$ (indexes will be omitted whenever possible). When $\mathsf{C} = \mathsf{Set}$, the set $\mathcal{N}_\omega$ can be taken to be the natural numbers which we denote $\mathbb{N}$.

In order to see how λ-presentable objects play the role of arities in an lλp category, note how they generate the category. From Definition 1.10, we know that every object is a λ-filtered colimit of λ-presentable objects; therefore we can define λ-accessible functors on the category by simply defining them on λ-presentable objects. The image on any other object will then be determined by the preservation of colimits, provided the target category is cocomplete.

This determines an adjoint equivalence $[\mathsf{C}_\lambda, \mathsf{D}] \cong [\mathsf{C}, \mathsf{D}]_\lambda$, between the category of functors from C_λ to D (and natural transformations between them) and the category of λ-accessible endofunctors between C and D. The functors in the two directions are precomposition with J_λ and extension along λ-filtered colimits. This is an instance of a *left Kan extension.*

Kan Extensions

Given a functor $I\colon \mathsf{A} \longrightarrow \mathsf{B}$ and a category C, precomposition with I defines a functor $-\circ I\colon [\mathsf{B}, \mathsf{C}] \longrightarrow [\mathsf{A}, \mathsf{C}]$. The problem of left and right Kan extensions is to find left and right adjoints to $- \circ I$. More concretely, given functors $F\colon \mathsf{A} \longrightarrow \mathsf{C}$ and $H\colon \mathsf{B} \longrightarrow \mathsf{C}$, the left and right Kan extensions satisfy the natural isomorphisms

$$[\mathsf{B}, \mathsf{C}](\mathsf{Lan}_I F, H) \cong [\mathsf{A}, \mathsf{C}](F, H \circ I) \qquad [\mathsf{B}, \mathsf{C}](H, \mathsf{Ran}_I F) \cong [\mathsf{A}, \mathsf{C}](H \circ I, F).$$

Kan extensions can be given pointwise using colimits and limits, or, more elegantly, using ends and coends (see [44, Chapter X] for details).

In the case of lλp categories, given a functor $\widetilde{F}\colon \mathsf{C}_\lambda \longrightarrow \mathsf{D}$, we can extend it along J_λ, and the action of its left Kan extension $F = \mathsf{Lan}_{J_\lambda} \widetilde{F}$ on an object X is given by the formula

$$FX = \int^{n \in \mathsf{C}_\lambda} \mathsf{C}(J_\lambda n, X) \otimes \widetilde{F} n, \tag{1.5}$$

where the operation $\otimes$ (often called *tensor*) is defined, for objects Y and Z in a category C and a set X, by the adjunction

$$\mathsf{C}(X \otimes Y, Z) \cong \mathsf{Set}(X, \mathsf{C}(Y, Z)). \tag{1.6}$$

The notation in (1.5) means that we form a diagram in D by taking the image of C_λ along the functor taking an object n to the $\mathsf{C}(J_\lambda n, X)$-fold coproduct of $\widetilde{F}n$ (that is what the action of $\otimes$ reduces to, here). The colimit of this diagram is the image of X along F, and the functor we get is then λ-accessible. Moreover, the adjunction above, in this case, defines an equivalence of categories:

$$\begin{tikzcd}[\mathsf{C}_\lambda, \mathsf{D}] \arrow[r, shift left=2, "\mathsf{Lan}_J"] \arrow[r, phantom, "\perp"] & {[\mathsf{C}, \mathsf{D}]_\lambda.} \arrow[l, shift left=2, "- \circ J"]\end{tikzcd} \tag{1.7}$$

In particular, this gives a very nice characterisation of accessible functors. A functor $F\colon \mathsf{C} \longrightarrow \mathsf{D}$ is λ-accessible if and only if it is the left Kan extension of its restriction to C_λ: $F = \mathsf{Lan}_{J_\lambda}(FJ_\lambda)$.

We can now revise the notion of signature with this new formalism. Recall how we presented a Set-signature as a map associating with each natural number the set of function symbols of that arity. We will then represent it as a map Σ from $\mathbb{N}$ to $|\mathsf{Set}|$. That is simply a functor from the category of finite Set-arities to Set itself. Given such a functor, finite terms are inductively defined by first constructing ground terms and then iterating the process countably many times. Leaving the iteration step aside for the time, we now focus on the functor F_Σ building ground terms. If f is a function symbol of arity n, we want to have a term $f(x_1, \ldots, x_n)$ for any choice of variables $x_1, \ldots, x_n$ from X. Such a choice is a function from n, thought of as a set, to X. For each such function, we want to get one term for each n-ary function symbol in the signature. In other words, we are taking the coproduct over n

of all sets of the form $\mathsf{Set}(n, X) \otimes \Sigma n$. Furthermore, because $\mathbb{N}$ is a discrete category, the coproduct is the same as a coend, and we can write

$$F_\Sigma X = \int^{n \in \mathbb{N}} \mathsf{Set}(n, X) \otimes \Sigma n. \tag{1.8}$$

The formula above is precisely that of (1.5), so we can more easily define the association $X \longmapsto F_\Sigma X$ as the functor $F_\Sigma = \mathsf{Lan}_{JI}\Sigma$. Functoriality clearly encodes variable renaming.

This construction generalises to any locally λ-presentable category C. We shall call any functor $\Sigma \colon \mathcal{N}_\lambda \longrightarrow \mathsf{C}$ a *signature*, and we shall write F_Σ for the λ-accessible endofunctor determined by the left Kan extension $\mathsf{Lan}_{JI}\Sigma$.

Example 1.15 (Monoids) We present here a signature $\Sigma_M \colon \mathbb{N} \longrightarrow \mathsf{Set}$ for monoids, leaving aside the equations, for the time being. We need a multiplication operation, which is a function symbol $\mathtt{m}$ of arity 2, and a unit for it. That is a constant, which we denote by $\mathtt{e}$. Hence, Σ_M will be defined as $\Sigma_M(0) = \{\mathtt{e}\}, \Sigma_M(2) = \{\mathtt{m}\}, \Sigma_M(n) = \emptyset$ for all other $n \in \mathbb{N}$.

Example 1.16 (Categories with $\top$) Also, categories with a terminal object can be seen as algebras for a specific theory, this time over Cat. This is an lfp category, and finitely presentable objects are finite colimits of finite categories. The signature $\Sigma_\top$ must declare the terminal object $\top$ and, for each object X, the unique map $!_X \colon X \longrightarrow \top$. Since the terminal object does not depend upon any data, its arity is the empty category $\mathsf{0}$. Since the map $!_X$ depends upon an object, its arity is the one object discrete category $\mathsf{1}$. Thus $\Sigma_\top(\mathsf{0}) = \mathsf{1}$, $\Sigma_\top(\mathsf{1}) = (\bullet \xrightarrow{!} \circ)$ (i.e. the category with two objects and one non-identity arrow) while $\Sigma_\top(c) = \mathsf{0}$ for any other finitely presentable category c. Notice that the two objects and the arrow in ($\bullet \xrightarrow{!} \circ$) are indeed term constructors. We denote the objects by different symbols because they are different term constructors. Equations will then set the source of the arrow defined above to an object X and the target to be $\top$.

1.2.2 Monoidal Categories

To recap, we have captured categorically the notion of arity and that of signature. Also, we have explained how, given a signature in a locally presentable category, we can determine an endofunctor on the category itself, which builds ground terms over it. In order to get all finite terms, we now need to iterate the process. This involves using composition of endofunctors, and properties of composition. These properties are perfectly expressed by the notion of a *monoidal category* (see [44, chapter VII] for more details).

Definition 1.17 A *monoidal category* is a category C together with a bifunctor $\otimes \colon \mathsf{C} \times \mathsf{C} \longrightarrow \mathsf{C}$ (called *tensor product*), and a *unit* I satisfying the following natural isomorphisms, expressing associativity, and the fact that I is a unit for $\otimes$:

$$\begin{aligned} \alpha &\colon \ \otimes \circ (\otimes \times \mathsf{Id}) \longrightarrow \otimes \circ (\mathsf{Id} \times \otimes), \\ \lambda &\colon \ (I \otimes -) \longrightarrow \mathsf{Id}, \\ \rho &\colon \ (- \otimes I) \longrightarrow \mathsf{Id}. \end{aligned}$$

These will have to make the following coherence diagrams commute for each

W, X, Y and Z in C:

$$\begin{tikzcd}
 & ((W \otimes X) \otimes Y) \otimes Z \arrow[dl, "\alpha\otimes\mathsf{id}"'] \arrow[dr, "\alpha"] & \\
(W \otimes (X \otimes Y)) \otimes Z \arrow[d, "\alpha"'] & & (W \otimes X) \otimes (Y \otimes Z) \arrow[d, "\alpha"] \\
W \otimes ((X \otimes Y) \otimes Z) \arrow[rr, "\mathsf{id}\otimes\alpha"'] & & W \otimes (X \otimes (Y \otimes Z))
\end{tikzcd}$$

$$\begin{tikzcd}
(X \otimes I) \otimes Y \arrow[rr, "\alpha"] \arrow[dr, "\rho\otimes\mathsf{id}"'] & & X \otimes (I \otimes Y) \arrow[dl, "\mathsf{id}\otimes\lambda"] \\
 & X \otimes Y &
\end{tikzcd}$$

A monoidal category is called *symmetric* if there is a family of isomorphisms $c_{XY}\colon X \otimes Y \longrightarrow Y \otimes X$ natural in X and Y which behaves well with respect to α, λ and ρ. When each functor $- \otimes Y$ has a right adjoint $[Y, -]$, so that

$$\mathsf{C}(X \otimes Y, Z) \cong \mathsf{C}(X, [Y, Z]),$$

the category is said to be *closed.*

Example 1.18 There are several examples of monoidal categories. We report here only those which will be pertinent to this thesis.

1. Any monoid, thought of as a discrete category, is clearly monoidal. The functors and the natural isomorphisms being the ones inherited from the monoidal structure. If the monoid is abelian, then it is symmetric as a monoidal category.

2. The category $\mathsf{End}(\mathsf{C})$ of endofunctors over a category C, whose objects are endofunctors and arrows are natural transformations between them,

is monoidal, with the tensor being functorial composition and the unit being the identity functor. Notice that, being composition associative and composition with identity ineffective, the natural isomorphisms of Definition 1.17 are all the identity. The category is clearly not symmetric, because composition of endofunctors is not commutative.

3. Given an lλp category C, the monoidal structure above restricts to a monoidal structure on $\mathsf{End}(\mathsf{C})_\lambda = [\mathsf{C}, \mathsf{C}]_\lambda$, essentially for the reason that λ-accessible functors are closed under composition. The induced structure transposes under the adjoint equivalence of (1.7) to give a monoidal structure on $[\mathsf{C}_\lambda, \mathsf{C}]$. This is defined by $F \otimes G = (\mathsf{Lan}_{J_\lambda} F) \circ G$. The unit for this tensor product is the inclusion functor J_λ. Again, this structure is neither symmetric nor closed.

Remark 1.19 Although it is not really relevant to our discussion, it is worth pointing out that the monoidal structure on the category $\mathsf{End}(\mathsf{C})$ is *strict*, i.e. the natural isomorphisms α, λ and ρ are all the identity. However, when we induce the monoidal structure on $\mathsf{End}(\mathsf{C})_\lambda$ we loose strictness, because the tensor product is defined via left Kan extensions and those are only unique up to isomorphic 2-cells.

Monoidal categories have enough structure to internalise the notion of monoid.

Definition 1.20 A *monoid* in a monoidal category C is an object M together with a pair of maps $\mu \colon M \otimes M \longrightarrow M$, $\eta \colon I \longrightarrow M$ such that the following diagrams commute:

$$\begin{tikzcd}
(M \otimes M) \otimes M \arrow[r, "\alpha"] \arrow[d, "\mu \otimes \mathsf{id}_M"'] & M \otimes (M \otimes M) \arrow[r, "\mathsf{id}_M \otimes \mu"] & M \otimes M \arrow[d, "\mu"] \\
M \otimes M \arrow[rr, "\mu"'] & & M
\end{tikzcd} \tag{1.9}$$

$$\begin{tikzcd}
I \otimes M \arrow[r, "\eta \otimes \mathsf{id}_M"] \arrow[dr, "\lambda"'] & M \otimes M \arrow[d, "\mu"'] & M \otimes I \arrow[l, "\mathsf{id}_M \otimes \eta"'] \arrow[dl, "\rho"] \\
& M &
\end{tikzcd} \tag{1.10}$$

A *monoid morphism* between monoids (M, η, μ) and (M', η', μ') is a map $\phi: M \longrightarrow M'$ such that $\phi\mu = \mu'(\phi \otimes \phi)$ and $\phi\eta = \eta'$.

Monoids and monoid morphisms form a subcategory of a monoidal category C, which we shall refer to as $\mathcal{M}on(\mathsf{C})$. There is an obvious forgetful functor $U: \mathcal{M}on(\mathsf{C}) \longrightarrow \mathsf{C}$, and given an object X in C, the *free monoid* M on X is a universal arrow $\phi: X \longrightarrow M$ from X to U; i.e. it is such that for every other monoid N and any other morphism $\psi: X \longrightarrow N$, there is a monoid morphism $\overline{\psi}: M \longrightarrow N$ such that $\overline{\psi}\phi = \psi$.

Example 1.21 A monoid in the category of sets, with the monoidal structure determined by cartesian products, is precisely a monoid in the classical sense. Also, the category Ab of abelian groups and group homomorphisms is cartesian (i.e. it has all finite products), hence it is monoidal. A monoid in this category is a unitary ring.

The notion of monoid in an endofunctor category is central to our work, but we shall explore it later.

1.2.3 Algebras and Models

In this section we introduce the notion of an algebra for a functor and relate it to that of a model for Σ, when the functor is of the form F_Σ.

Recall that, if Σ is a Set-signature, a model for it is given by a set A and, for any n-ary function symbol f, a function $[\![f]\!]\colon A^n \longrightarrow A$ interpreting it. Let's focus our attention on the interpreting maps. They form a collection of maps over the same object A. Using the universal property of coproducts, these induce a function

$$[\![\Sigma]\!]_A\colon \coprod_{f\in\Sigma} A^{\mathsf{ar}(f)} \longrightarrow A.$$

It's not hard to see that the source of $[\![\Sigma]\!]_A$ is the action of F_Σ on X, given that arities form a discrete category. Hence, we can conveniently encode all the relevant structure of a model as a map from $F_\Sigma A$ to A.

Conversely, given any map of such form, we can recover a model for the signature by simply splitting it along the different components of the coproduct. Furthermore, a model homomorphism ϕ between two models $(A, [\![\ ]\!]_A)$ and $(B, [\![\ ]\!]_B)$ determines a commutative diagram

$$\begin{tikzcd}
F_\Sigma A \arrow[r, "F_\Sigma \phi"] \arrow[d, "{[\![\Sigma]\!]_A}"'] & F_\Sigma B \arrow[d, "{[\![\Sigma]\!]_B}"] \\
A \arrow[r, "\phi"'] & B,
\end{tikzcd}$$

and conversely any such diagram determines a model homomorphism.

This defines an isomorphism between the category of models for a theory and another category: that of F_Σ-*algebras.*

Definition 1.22 Given an endofunctor F on a category C, the category $F-\mathsf{Alg}$ of *F-algebras* has as objects pairs (A,α), where $\alpha: FA \longrightarrow A$ is an arrow in C. A map in $F-\mathsf{Alg}$ between two algebras (A,α) and (B,β), called an *F-algebra homomorphism*, is a map $\phi: A \longrightarrow B$ such that the following square commutes:

$$\begin{tikzcd}
FA \arrow[r, "F\phi"] \arrow[d, "\alpha"'] & FB \arrow[d, "\beta"] \\
A \arrow[r, "\phi"'] & B.
\end{tikzcd}$$

The *forgetful functor* $U: F_\Sigma - \mathsf{Alg} \longrightarrow \mathsf{C}$ maps an algebra (A,α) to its *carrier* A and an algebra homomorphism ϕ to the map itself.

Although it is a folklore result, we give here a proof of the following proposition, as it seems to be lacking in the literature.

Proposition 1.23 *Let F be an endofunctor on a category C. Then, the forgetful functor $U: F-\mathsf{Alg} \longrightarrow \mathsf{C}$ creates:*

1. *all limits which exist in C;*

2. *all colimits which exist in C and are preserved by F.*

Proof.

1. Let $X: \mathsf{D} \longrightarrow \mathsf{C}$ be a diagram in C, such that each object X_d has an algebra structure ϕ_d and each morphism $X_f: X_d \longrightarrow X_{d'}$ is an algebra morphism, and put $Y = \lim X$, with projections $p_d: Y \longrightarrow X_d$. Then, the collection of morphisms $FY \xrightarrow{Fp_d} FX_d \xrightarrow{\phi_d} X_d$ determines a map

from FY to Y such that each ϕ_d is an algebra morphism. This is the limit of the diagram X in $F-\mathsf{Alg}$, as easily checked.

2. For colimits the argument needs the preservation of them by F. Given the same diagram X, and given its colimit Z, we get a map from $\operatorname{colim} FX_d$ to Z induced by the cocone having as maps the algebras ϕ_d. Now, because F preserves the colimit, we have the algebra structure $FZ \cong \operatorname{colim} FX_d \longrightarrow Z$, and again it is easy to see that this is the colimit of the diagram.

□

As we have seen above, for any finitary signature Σ on Set the category of F_Σ-algebras is isomorphic to the category of models for the theory consisting of Σ and no equations. It is therefore natural to consider the category of F_Σ-algebras as the *category of models* for any signature in any lλp category, and this we shall do henceforth.

Amongst all algebras for an endofunctor, a particular role is played by the initial one, because its universal property matches exactly the induction principle, as we understand by studying its construction.

This is indeed an instance of a very general construction, the most comprehensive discussion of which is in a paper by Max Kelly [34]. We highlight here the construction as it is usually presented.

Let F be a λ-accessible endofunctor on an lλp category C with initial object I. The *initial F-algebra chain* in C is the chain $\mathcal{A}$

$$A_0 \xrightarrow{\phi_{0,1}} A_1 \xrightarrow{\phi_{1,2}} \cdots \xrightarrow{\phi_{i,\mu}} A_\mu \xrightarrow{\phi_{\mu,j}} \cdots \tag{1.11}$$

which is defined by induction as follows. The objects will be

$$\begin{aligned} A_0 &= I \\ A_{n+1} &= F(A_n) \quad \text{for any non-limit ordinal } n \\ A_\mu &= \operatorname{colim} D \quad \text{for a limit ordinal } \mu \end{aligned}$$

where D is the chain constructed until that point, i.e. containing all the A_i with $i < \mu$. Maps between them will be

$$\begin{aligned} \phi_{0,1}\colon A_0 \longrightarrow A_1 &= \text{the unique arrow from } I \\ \phi_{i,j}\colon A_i \longrightarrow A_j &= F(\phi_{i-1,j-1}) \text{ for non-limit ordinals } i < j \\ \phi_{i,\mu}\colon A_i \longrightarrow A_\mu &= \text{the colimiting map for a limit ordinal } \mu > i \\ \phi_{\mu,j}\colon A_\mu \longrightarrow A_j &= \text{the map determined by the collection } (\phi_{l,j})_{l<\mu} \\ & \quad\ \text{for a limit cardinal } \mu < j \end{aligned}$$

Lemma 1.24 *If the initial F-algebra chain (1.11) converges at the step λ (i.e. if $A_\lambda \cong A_{\lambda+1}$, where λ is any ordinal), then $(A, \phi^{-1}_{\lambda,\lambda+1})$ is initial in $F-\mathsf{Alg}$.*

For a proof of this result, see [3]. In [9, Proposition IV.2.5] the following result is proved, providing a sufficient condition for convergence of the chain.

Lemma 1.25 *Let F be λ-accessible. Then the initial F-algebra chain converges within λ many steps.*

Putting the two results together, one gets the fundamental corollary:

Proposition 1.26 *Every λ-accessible endofunctor on an $l\lambda p$ category has an initial algebra.*

In order to understand how this construction models the classical inductive definition of terms, let's consider a finitary signature Σ on Set and work out the initial F_Σ-algebra T_Σ. Being the functor finitary, we know the chain will terminate in ω steps. The set A_0 will be the initial object, that is the empty set. $A_1 = F_\Sigma A_0$ is the set consisting of all terms of depth one built over Σ but involving no variable. In other words, we get just the constants. The unique map from A_0 to A_1 is the empty function. A_2 is the set of terms of depth one built over the Σ with variables in A_1, i.e. terms of depth at most two. The map from A_1 to A_2 is the inclusion of constants. More generally, A_n is the set of terms of depth at most n, and the function $\phi_{n,m}\colon A_n \longrightarrow A_m$, for $n < m$, is the obvious inclusion. The colimit of the chain is then clearly the set $T_\Sigma\emptyset$ of all closed terms of finite depth built over Σ. The inductive clauses defining $T_\Sigma\emptyset$ as in (1.1) are matched here by the chain in (1.11). The initiality of this algebra is a counterpart to the definition of functions on terms by structural induction.

Although in this setting we get it for free by Lemma 1.24, it's worth pointing out one of the main properties of initial algebras of an endofunctor on any category.

Lemma 1.27 (Lambek's lemma) *If $f\colon FT \longrightarrow T$ is the initial algebra of an endofunctor F, then f is an isomorphism.*

The proof [41] generalises to any category the argument used to show that the least prefixed point of a monotone function on a partial order is indeed a fixpoint.

Variables and Substitution

With the theoretical machinery we have developed so far, we can easily model notions like variable and substitution. The usual way of proceeding, on a syntactic level, is to consider variables as constants, and build finite terms over the resulting enriched signature (as in (1.1)). Here we do the same. Let's recall that the endofunctor for a finitary signature Σ on Set evaluates as in (1.5), or equivalently as

$$F_\Sigma Y = \Sigma_0 + \coprod_{n>0} \Sigma_n \times Y^n.$$

Enriching the signature with a set of variables X, amounts to replacing Σ_0 with $\Sigma_0 + X$ above. Therefore, it is the same as considering the endofunctor $X + F_\Sigma$ (where we now write X for the constant functor K_X).

The initial algebra for this functor has carrier the set $T_\Sigma X$ of finite Σ-terms over X, and its algebra map is $[\eta_X, \alpha_X]\colon X + FT_\Sigma X \longrightarrow T_\Sigma X$. The first component of the map is just including the variables into the set of terms, whereas α_X shows that these terms for a model for Σ. Notice also that, given any other F_Σ-algebra (B, β) and any map $\sigma\colon X \longrightarrow B$, we naturally have an $X + F_\Sigma$-algebra with carrier B and structure $[\sigma, \beta]$. Hence, the initiality of $T_\Sigma X$ determines an algebra morphism $\widehat{\sigma}$ such that

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[dr, "\sigma"'] & T_\Sigma X \arrow[d, dashed, "\widehat{\sigma}"'] & F_\Sigma T_\Sigma X \arrow[l, "\alpha"'] \arrow[d, "F_\Sigma \widehat{\sigma}"] \\
 & B & F_\Sigma B \arrow[l, "\beta"]
\end{tikzcd} \tag{1.12}$$

commutes. Syntactically, the map $\widehat{\sigma}$ is defined again by induction on the structure of terms: σ is the base of the induction, whereas β is the inductive step.

Diagram (1.12) expresses the fact that T_Σ is the free F_Σ-algebra over X, i.e. η_X is universal from X to the forgetful functor $U\colon F_\Sigma - \mathsf{Alg} \longrightarrow \mathsf{C}$. When the initial $X + F_\Sigma$-algebra exists for every X, these universal arrows define a left adjoint to U. When this is the case, the association $X \longmapsto (T_\Sigma X, \alpha_X)$ is functorial. The unit of the adjunction is the natural transformation $\eta\colon \mathsf{Id} \longrightarrow T_\Sigma$.

By the properties of initiality, we can also determine another natural transformation from T_Σ^2 to T_Σ. $T_\Sigma^2 X$ is the free F_Σ-algebra on $T_\Sigma X$, which is itself an F_Σ -algebra. The identity on T_Σ, therefore, induces an algebra morphism $\mu_X\colon T_\Sigma^2 X \longrightarrow T_\Sigma X$. Using the properties of initiality, one can easily show that this collection of maps is natural, and moreover, the following diagrams commute:

$$\begin{tikzcd}
T_\Sigma \arrow[r, "\eta T_\Sigma"] \arrow[dr, "\mathsf{id}"'] & T_{\Sigma^2} \arrow[d, "\mu"] & T_\Sigma \arrow[l, "T_\Sigma \eta"'] \arrow[dl, "\mathsf{id}"] \\
 & T_\Sigma &
\end{tikzcd}
\qquad
\begin{tikzcd}
T_\Sigma^3 \arrow[r, "T_\Sigma \mu"] \arrow[d, "\mu T_\Sigma"'] & T_\Sigma^2 \arrow[d, "\mu"] \\
T_\Sigma^2 \arrow[r, "\mu"'] & T_\Sigma
\end{tikzcd} \tag{1.13}$$

The structure arising on T_Σ is well known in category theory under the name of *monad.* Monads give an account for the notions of substitution and variables. For this reason, we shall focus on them for the rest of this thesis.

1.2.4 Monads and Algebras

Mac Lane's book [44] introduces the notion of monad, as a triple (T, η, μ) consisting of an endofunctor T on a category C, together with two natural transformations $\eta\colon \mathsf{Id} \longrightarrow T$ and $\mu\colon T^2 \longrightarrow T$ making the two diagrams in (1.13) commute. An exhaustive treatment of the 2-categorical aspects of the subject is given by Ross Street [55], but we shall not take that approach here. For us, a monad will be described by any of the following equivalent presentations.

Theorem 1.28 *On a category C, the following assignations are equivalent:*

a) (Eilenberg-Moore) an endofunctor T on C together with two natural transformations $\eta\colon \mathsf{Id} \longrightarrow T$ and $\mu\colon T^2 \longrightarrow T$ making (1.13) commute;

b) (Kleisli) an endofunction T on $|\mathsf{C}|$, for each $X \in |\mathsf{C}|$ an arrow in C $\eta_X\colon X \longrightarrow TX$, and, for any pair of objects X and Y in C, a map $\mathsf{s}_{X,Y}\colon \mathsf{C}(X, T(Y)) \longrightarrow \mathsf{C}(T(X), T(Y))$, such that the following properties are satisfied for all X, Y and Z in $|\mathsf{C}|$, $f \in \mathsf{C}(X, T(Y))$ and $g \in \mathsf{C}(Y, T(Z))$:

1. $\mathsf{s}_{X,X}(\eta_X) = \mathsf{id}_{T(X)}$;

2. $\mathsf{s}_{X,Y}(f)\eta_X = f$;

3. $\mathrm{s}_{Y,Z}(g)\mathrm{s}_{X,Y}(f) = \mathrm{s}_{X,Z}(\mathrm{s}_{Y,Z}(g)f)$*;*

c) (Monoid) a monoid (T, η, μ) *on the monoidal category* $\mathsf{End}(\mathsf{C})$.

Proof.

a) $\Longleftrightarrow$ *c)* Let's recall from Example 1.18-2 that $\mathsf{End}(\mathsf{C})$ is a monoidal category, the tensor being defined by composition. The diagrams in (1.13) now translate exactly in (1.10) and (1.9) from Definition 1.20.

a) $\Longleftrightarrow$ *b)* Given a triple (T, η, μ) as in *a*), we define the action of $\mathrm{s}_{X,Y}$ on $f\colon X \longrightarrow TY$ as the composite $\mathrm{s}_{X,Y}(f) = \mu_Y T(f)\colon TX \longrightarrow TY$. Equations 1, 2 and 3 are satisfied because of the naturality and functoriality of T, η and μ.

Conversely, given T, η and s as in *b*), we define the action of T on an arrow $f\colon X \longrightarrow Y$ as $T(f) = \mathrm{s}_{X,Y}(\eta_Y f)$. This makes T into a functor and η into a natural transformation, as one can easily check. The X-th component of μ is defined as $\mu_X = \mathrm{s}_{X,X}(\mathsf{id}_{TX})\colon T^2X \longrightarrow TX$. Naturality is again easily proved.

□

Definition 1.29 A *monad* T on a category C is given by any assignation as in Theorem 1.28. When referring to the Kleisli presentation, we shall omit the subscript to the function s whenever possible, and will occasionally refer to the monad as a *Kleisli triple.* Following Definition 1.20, we shall refer to commutativity of the triangles in (1.13) as *unit laws*, whereas commutativity of the square is the *multiplication law.*

Remark 1.30 The notion of monoid morphism introduced in Definition 1.20 translates under Theorem 1.28 into the notion of *monad morphism.* Given monads (T, η, μ) and (T', η', μ'), a monad morphism ϕ between them will be a natural transformation $\phi: T \longrightarrow T'$ such that $\phi\eta = \eta'$ and $\mu'\phi^2 = \phi\mu$.

In the language of Kleisli triples, a morphism is an indexed family of maps $\phi_X: TX \longrightarrow T'X$ such that

1. $\phi_X \eta_X = \eta'_X$;
2. $\phi_Y \mathsf{s}(f) = \mathsf{s}'(\phi_Y f)\phi_X$.

Definition 1.31 We shall denote by $\mathsf{Mon}(\mathsf{C})$ the *category of monads over* C. This has monads as objects and monad morphisms as arrows, and it is clearly isomorphic to the category of monoids in the monoidal category $\mathsf{End}(\mathsf{C})$. There is an obvious forgetful functor $V: \mathsf{Mon}(\mathsf{C}) \longrightarrow \mathsf{End}(\mathsf{C})$ assigning to each monad its underlying endofunctor. We shall write $\mathsf{Mon}(\mathsf{C})_\lambda$ for the full subcategory of $\mathsf{Mon}(\mathsf{C})$ consisting of those monads whose underlying endofunctor is λ-accessible, and call its objects *λ-accessible monads.* The forgetful functor restricts to a functor $V_\lambda: \mathsf{Mon}(\mathsf{C})_\lambda \longrightarrow \mathsf{End}(\mathsf{C})_\lambda$.

Remark 1.32 As we saw in Example 1.18-3, the monoidal structure of $\mathsf{End}(\mathsf{C})_\lambda$ transposes under equivalence (1.7) to a monoidal structure on $[\mathsf{C}_\lambda, \mathsf{C}]$. It turns out that the equivalence can then be restricted to the category of monoids and monoid morphisms on both sides, hence inducing an equivalence between $\mathsf{Mon}(\mathsf{C})_\lambda$ and the category $\mathcal{M}on([\mathsf{C}_\lambda, \mathsf{C}])$.

When considering algebras over the underlying functor of a monad T, it is natural to require that they respect the existing additional structure.

Definition 1.33 Let $\mathsf{T} = (T, \eta, \mu)$ be a monad on C. The *category* $\mathsf{T} - \mathsf{Alg}$ *of* T*-algebras*, also called the *Eilenberg-Moore category of* T, has as objects those T-algebras $\alpha \colon TX \longrightarrow X$ for which the following commute:

$$\begin{tikzcd}
X \arrow[r, "\eta_X"] \arrow[dr, "\mathsf{id}"'] & TX \arrow[d, "\alpha"] & T^2X \arrow[r, "\mu"] \arrow[d, "T\alpha"'] & TX \arrow[d, "\alpha"] \\
& X & TX \arrow[r, "\alpha"'] & X.
\end{tikzcd} \tag{1.14}$$

Commutativity of the left triangle is called the *unit identity*, whereas that of the right square is called the *multiplication identity*. Maps between two algebras $\alpha \colon T(X) \longrightarrow X$ and $\beta \colon T(Y) \longrightarrow Y$, called T*-algebra homomorphisms*, are those arrows $f \colon X \longrightarrow Y$ in C such that $f\alpha = \beta T(f)$. Composition and identities are those in C.

Monads are intimately related to adjunctions [44, 55]. Given a pair of adjoint functors

$$\begin{tikzcd}
\mathsf{C} \arrow[r, shift left, "F"] \arrow[r, phantom, "\perp"] & \mathsf{D} \arrow[l, shift left, "G"]
\end{tikzcd} \tag{1.15}$$

one can always determine a monad by taking the triple $(GF, \eta, G\varepsilon F)$, where η and ε are the unit and counit of the adjunction. Conversely, given any monad (T, η, μ) on a category C, one can build an adjunction

$$\begin{tikzcd}
\mathsf{C} \arrow[r, shift left, "T"] \arrow[r, phantom, "\perp"] & \mathsf{T} - \mathsf{Alg} \arrow[l, shift left, "U"]
\end{tikzcd}$$

where the left adjoint, called T with an abuse of notation, maps the object X to the free T-algebra $\mu_X \colon T^2X \longrightarrow TX$ over it, and U is the forgetful functor. If we now consider the monad arising from this adjunction, we get T again. In fact, $\mathsf{T} - \mathsf{Alg}$ is final amongst all categories with an adjoint pair of functors over C giving rise to the monad T and related by functors commuting with the right adjoints. That is to say that, whenever there is another category D and an adjoint pair as in (1.15) such that $GF = T$, there is a unique functor $\phi\colon \mathsf{D} \longrightarrow \mathsf{T}-\mathsf{Alg}$ such that $U\phi = G$. We say that D is *monadic* over C if ϕ is an equivalence. When ϕ is full and faithful, we say that the functor G is *of descent type.*

Kelly and Power [37] showed the equivalence of the following facts for an adjunction $F \dashv G\colon \mathsf{C} \longrightarrow \mathsf{D}$, which will prove very useful later on:

1. G is of descent type;

2. the counit of the adjunction is pointwise a coequaliser;

3. each algebra for the monad $T = FG$ is the coequaliser of a parallel pair of morphisms between two free algebras.

A proof of the following can be found in [12, Prop. 5.2]:

Proposition 1.34 *Let F and $\mathsf{T} = (T, \eta, \mu)$ be respectively an endofunctor and a monad on a category C. Then, there is a bijection between natural transformations from F to T and functors from $\mathsf{T}-\mathsf{Alg}$ to F−Alg respecting the forgetful functors:*

$$\begin{tikzcd}
\mathsf{T}-\mathsf{Alg} \arrow[rr] \arrow[dr, "U_T"'] & & F\text{−}\mathsf{Alg} \arrow[dl, "U_F"] \\
& \mathsf{C} &
\end{tikzcd}$$

If, moreover, F is the functor part of a monad $\mathsf{F} = (F, \eta', \mu')$, then the functor restricts to the category of F-algebras if and only if the natural transformation is indeed a monad morphism.

We can now explain how the notions of monad and algebra for a monad model the properties of terms. Recall that, if Σ is a finitary signature on Set, then $T_\Sigma X$ is the set of finite terms over Σ with variables in X, i.e. the free F_Σ-algebra on X. The natural transformation η_X maps each variable x to the term consisting of x itself. Applying T_Σ twice builds terms whose variables are terms themselves. The natural transformation μ *flattens* terms by performing a substitution. For example, if we consider the term $\mathtt{F}(t_1, t_2)$ in $T_\Sigma T_\Sigma\{x, y\}$ where $t_1 = x$ and $t_2 = \mathtt{G}(y)$, then the action of μ on this term maps it to $\mathtt{F}(x, \mathtt{G}(y)) \in T_\Sigma\{x, y\}$. Diagrams (1.13) express the desirable coherence properties of substitution, namely the fact that flattening respects variables and that it is associative.

One of the key points of denotational semantics is that, given an interpretation of an n-ary symbol in a signature Σ as a function from A^n to A (where A is a model of Σ), we can inductively define an interpretation of each finite term as a function taking as many arguments as the different variables appearing in it. The set $T_\Sigma A$ can be thought of as the set of formal applications of a term (thought of as a function) to elements in the model A. Performing such application, we determine a map $\alpha: T_\Sigma A \longrightarrow A$. The diagrams in (1.14) express the coherence of the action of performing the application with respect to the inner structure of terms.

Free Monads

Although the intuition we have explained is intrinsic to the fact that we work over Set, we made no explicit use of its properties. In fact, this whole theory can be extended to many other categories. All we need is a context where to speak of arities and signatures; hence, we focus on locally presentable categories. Furthermore, we shall often start with a λ-accessible endofunctor F on an lλp category C, the functors F_Σ arising from a signature being just a special case.

The forgetful functor $U\colon F-\mathsf{Alg} \longrightarrow \mathsf{C}$ clearly reflects isomorphisms, and by Proposition 1.23 it creates coequalisers of U-contractible coequaliser pairs, since – being absolute – they are preserved by F. We also know by Proposition 1.26 that, given X in C, we can form the free F-algebra on it, whose carrier is $T_F X$, and this defines a left adjoint to U. The adjunction gives rise to a monad T_F, and by Beck's Precise Tripleability Theorem [14] the category of F-algebras is isomorphic to $\mathsf{T}_F - \mathsf{Alg}$. Under Proposition 1.34, the isomorphism induces a natural transformation $\zeta\colon F \longrightarrow T_F$.

Let now $\alpha\colon F \longrightarrow T$ be a natural transformation between a λ-accessible endofunctor F and a λ-accessible monad T. Then, by Proposition 1.34, there is a functor from $\mathsf{T}-\mathsf{Alg}$ to F–Alg which, under the isomorphism, determines a functor from $\mathsf{T}-\mathsf{Alg}$ to $\mathsf{T}_F-\mathsf{Alg}$. This, in turn, induces a monad morphism $\widehat{\alpha}\colon \mathsf{T}_F \longrightarrow \mathsf{T}$, which is the unique one such that $\widehat{\alpha}\zeta = \alpha$.

This is just one possible way to show that T_F is the *free monad* over F.

Definition 1.35 Let C be a category, and F an endofunctor on it. The *free monad* over F is a universal arrow from F to the forgetful functor

$V\colon \mathsf{Mon}(\mathsf{C}) \longrightarrow \mathsf{End}(\mathsf{C})$.

Depending on how we choose to present the free monad, we clearly get different means of proving its freeness. For example, we could work in the monoidal category of λ-accessible endofunctors (the tensor being given by composition) and, for a given F, consider the endofunctor $\mathsf{Id}+F\circ -$, mapping each endofunctor G to $\mathsf{Id}+FG$. The initial algebra chain (1.11) now converges to an endofunctor T_F [34], which is the underlying object of the free monoid over F, i.e. the free monad over the endofunctor. Note that the chain will now stop within λ many steps.

In general, the free monad on a functor might not even exist; it does however, by Lemma 1.24, when C is lλp and F is λ-accessible. The following result collects some of the equivalent ways of presenting the free monad over an endofunctor F. Its proof can be recollected from [44, 34, 37, 14].

Proposition 1.36 *Let F be a λ-accessible endofunctor over an lλp-category C. Then, any of the following definitions gives rise to the free monad T_F over F:*

1. *For every X in C, $T_F X$ is the carrier of the initial $X+F$-algebra.*
2. *$T_F \cong UL$, where L is the left adjoint to $U\colon F{-}\mathsf{Alg} \longrightarrow \mathsf{C}$.*
3. *T_F is the colimit of the initial algebra chain (1.11) for the endofunctor $\mathsf{Id}+F\circ -$ on $\mathsf{End}(\mathsf{C})_\lambda$*
4. *T_F is the free monoid over F in $\mathsf{End}(\mathsf{C})_\lambda$.*

By definition, the free monad is a universal arrow. It follows that, if we can form it for any endofunctor on a category C, we get a left adjoint to the forgetful functor $V\colon \mathsf{Mon}(\mathsf{C}) \longrightarrow \mathsf{End}(\mathsf{C})$.

Therefore, when C is lλp, every λ-accessible endofunctor admits by Proposition 1.26 a free monad over itself, and we get the following chain of adjunctions, relating signatures and monads:

$$\begin{tikzcd}
{[\mathcal{N}_\lambda, \mathsf{C}]} \arrow[r, shift left, "\mathsf{Lan}_{I_\lambda}"] \arrow[r, phantom, "\perp"] & {[\mathsf{C}_\lambda, \mathsf{C}]} \arrow[l, shift left, "- \circ I_\lambda"] \arrow[r, shift left, "\mathsf{Lan}_{J_\lambda}"] \arrow[r, phantom, "\cong"] & \mathsf{End}(\mathsf{C})_\lambda \arrow[l, shift left, "- \circ J_\lambda"] \arrow[r, shift left, "H_\lambda"] \arrow[r, phantom, "\perp"] & \mathsf{Mon}(\mathsf{C})_\lambda \arrow[l, shift left, "V_\lambda"]
\end{tikzcd} \tag{1.16}$$

In particular, it follows that, given a signature Σ of rank λ, there is a free monad T_Σ on it. Recall that the category $F_\Sigma\!-\!\mathsf{Alg}$ is isomorphic to the category of models for Σ. Now we also know that these are isomorphic to the category of algebras for the monad T_Σ, and the properties of the monad give us a better way of handling variables and substitution.

In fact, the picture can be completed by adding more elements. Recall from Remark 1.32 how equivalence (1.7) restricts to an equivalence

$$\begin{tikzcd}
\mathcal{M}on([\mathsf{C}_\lambda, \mathsf{C}]) \arrow[r, shift left, "\mathsf{Lan}_{J_\lambda}"] \arrow[r, phantom, "\perp"] & \mathsf{Mon}(\mathsf{C})_\lambda . \arrow[l, shift left, "- \circ J_\lambda"]
\end{tikzcd}$$

The adjunctions above can then be presented again in the following way.

$$\begin{tikzcd}
{[\mathcal{N}_\lambda, \mathsf{C}]} \arrow[r, shift left, "\mathsf{Lan}_{I_\lambda}"] \arrow[r, phantom, "\perp"] & {[\mathsf{C}_\lambda, \mathsf{C}]} \arrow[l, shift left, "- \circ I_\lambda"] \arrow[r, shift left] \arrow[r, phantom, "\perp"] \arrow[d, shift left, "\mathsf{Lan}_{J_\lambda}"] \arrow[d, phantom, "\cong"] & \mathcal{M}on([\mathsf{C}_\lambda, \mathsf{C}]) \arrow[l, shift left] \arrow[d, shift left, "\mathsf{Lan}_{J_\lambda}"] \arrow[d, phantom, "\cong"] \\
& \mathsf{End}(\mathsf{C})_\lambda \arrow[u, shift left, "- \circ J_\lambda"] \arrow[r, shift left, "H_\lambda"] \arrow[r, phantom, "\perp"] & \mathsf{Mon}(\mathsf{C})_\lambda \arrow[l, shift left, "V_\lambda"] \arrow[u, shift left, "- \circ J_\lambda"]
\end{tikzcd} \tag{1.17}$$

Remark 1.37 When we shall come to the dual of this setting, in Section 2.3, a construction like the above will not be possible since one would be interested in the limit of a co-chain, and there is no reason for accessible endofunctors to preserve such limits. As an effect of this, the rank of the cofree comonad over an accessible endofunctor may increase. This change of rank underlies the technical difficulties which will arise in Section 2.3.

1.2.5 Adding Equations

In the previous sections we generalised the notion of signature to locally presentable categories, and we showed how a signature gives rise to a monad such that its Eilenberg-Moore category is equivalent to its category of models. Now we turn our attention to algebraic theories.

In order to model this notion categorically, it is useful to think of equations as terms which prove the equality of two terms over the signature. Given an algebraic theory $\langle \Sigma, E \rangle$, we can consider E as a *signature of equations*, giving, for any arity n, the set of equations between terms which are built on n variables. The arity of an equation $X \vdash t = s$ is the cardinality of the set X.

Let's consider, as an example, the theory of monoids, whose signature we introduced in Example 1.15. We need to impose three equations: left unit, right unit (both unary) and associativity (ternary). Hence we set $E(3) = \{a\}$, $E(1) = \{l, r\}$ and $E(n) = \emptyset$ for any other n. Intuitively, the term $a(x, y, z)$ "proves" that $\mathtt{m}(\mathtt{m}(x, y), z) = \mathtt{m}(x, \mathtt{m}(y, z))$, whereas $l(x)$ is a proof of the fact that $\mathtt{m}(\mathtt{e}, x) = x$ and similarly for $r(x)$.

Given a signature of equations, we associate to each equation symbol its left and right handsides by means of pairs of parallel arrows λ'_n and ρ'_n from

$E(n)$ to $\mathsf{T}_\Sigma(n)$. In the case above we get:

$$\begin{array}{rclrclrcl}
\lambda'_1(l) & = & \mathtt{m}(\mathtt{e},x) & \lambda'_1(r) & = & \mathtt{m}(x,\mathtt{e}) & \lambda'_3(a) & = & \mathtt{m}(\mathtt{m}(x,y),z) \\
\rho'_1(l) & = & x & \rho'_1(r) & = & x & \rho'_3(a) & = & \mathtt{m}(x,\mathtt{m}(y,z))
\end{array}$$

This way, we determine two natural transformations:

$$\begin{tikzcd} E \arrow[r, shift left, "\lambda'"] \arrow[r, shift right, "\rho'"'] & \mathsf{T}_\Sigma JI : \mathbb{N} \arrow[r] & \mathsf{Set} \end{tikzcd}$$

$\mathsf{T}_\Sigma JI$ is to be thought of as the signature which has as n-ary symbols all Σ-terms over n variables. By the composite adjunction of (1.16), λ' and ρ' induce two monad morphisms from T_E to T_Σ. Taking their coequaliser q in $\mathsf{Mon(C)}_\lambda$, we get a monad whose algebras are precisely those algebras for T_Σ satisfying all the equations in E [34]:

$$\begin{tikzcd} \mathsf{T}_E \arrow[r, shift left, "\lambda"] \arrow[r, shift right, "\rho"'] & \mathsf{T}_\Sigma \arrow[r, "q"] & \mathsf{T}_{\langle\Sigma,E\rangle}. \end{tikzcd} \tag{1.18}$$

Kelly and Power [37] proved that the composite $V_\lambda(-)J_\lambda I_\lambda$ in (1.16) is of descent type, thus inducing, because of the equivalent facts stated on page 42, the following representation theorem.

Theorem 1.38 *Any λ-accessible monad over an $l\lambda p$ category C is a coequaliser of two free monads.*

This is saying precisely that every λ-accessible monad is modelling an algebraic theory consisting of a signature and some equations.

However, one should be aware of the fact that the signature and the equations we get for a monad T do not necessarily give an idea of the theory we are considering. In fact, the way the signature is reconstructed is just by precomposing T with the inclusion JI, therefore, as mentioned above, we get as term constructors of arity n all the elements of the free algebra of terms in n variables, but this does not show what the key constructors of the theory really are.

1.2.6 Introducing Enrichment

In this last section of the Chapter we introduce the notion of enriched category, and put it to use in order to confer more generality to the theory we developed so far.

An enriched category is a category where the hom-sets are replaced by objects of a monoidal category. We recall here the basic definitions of $\mathcal{V}$-category, $\mathcal{V}$-functor and $\mathcal{V}$-natural transformation, taking them from Kelly's book and referring to it for any result on the subject [35].

Let's fix a monoidal category $(\mathcal{V}, \otimes, I)$, with the natural isomorphisms α, λ, ρ.

Definition 1.39 A (small) *enriched category* C over $\mathcal{V}$ is specified by a collection of *objects* C_0; for each pair X, Y of objects in C_0, an object in $\mathcal{V}$, which we denote by $\mathsf{C}(X,Y)$; for each object X in C_0 a map $j_X: I \longrightarrow \mathsf{C}(X,X)$ in $\mathcal{V}$, and, for objects X, Y and Z, a $\mathcal{V}$-map $M_{XYZ}: \mathsf{C}(X,Y) \otimes \mathsf{C}(Y,Z) \longrightarrow \mathsf{C}(X,Z)$ such that the following diagrams commute:

$$\begin{tikzcd}
(\mathsf{C}(X,Y) \otimes \mathsf{C}(Y,Z)) \otimes \mathsf{C}(Z,W) \arrow[rr, "\alpha"] \arrow[d, "M\otimes\mathsf{id}"'] & & \mathsf{C}(X,Y) \otimes (\mathsf{C}(Y,Z) \otimes \mathsf{C}(Z,W)) , \arrow[d, "\mathsf{id}\otimes M"] \\
\mathsf{C}(X,Z) \otimes \mathsf{C}(Z,W) \arrow[r, "M"'] & \mathsf{C}(X,W) & \mathsf{C}(X,Y) \otimes \mathsf{C}(Y,W) \arrow[l, "M"]
\end{tikzcd}$$

$$\begin{tikzcd}
I \otimes \mathsf{C}(X,Y) \arrow[d, "j\otimes\mathsf{id}"'] \arrow[dr, "\lambda"] & & \mathsf{C}(X,Y) \otimes I \arrow[d, "\mathsf{id}\otimes j"] \arrow[dl, "\rho"'] \\
\mathsf{C}(X,Y) \otimes \mathsf{C}(Y,Y) \arrow[r, "M"'] & \mathsf{C}(X,Y) & \mathsf{C}(X,X) \otimes \mathsf{C}(X,Y). \arrow[l, "M"]
\end{tikzcd}$$

Definition 1.40 Given two $\mathcal{V}$-enriched categories C and D, a *$\mathcal{V}$-enriched functor F* between them is a pair consisting of a function $F\colon \mathsf{C}_0 \longrightarrow \mathsf{D}_0$ and, for any pair X, Y of objects in C, a $\mathcal{V}$-map $F_{XY}\colon \mathsf{C}(X,Y) \longrightarrow \mathsf{D}(FX,FY)$ such that the following diagrams commute:

$$\begin{tikzcd}
\mathsf{C}(X,Y) \otimes \mathsf{C}(Y,Z) \arrow[r, "M"] \arrow[d, "F_{XY}\otimes F_{YZ}"'] & \mathsf{C}(X,Z) \arrow[d, "F_{XZ}"] & I \arrow[r, "j"] \arrow[dr, bend right, "j"'] & \mathsf{C}(X,X) \arrow[d, "F_{XX}"] \\
\mathsf{D}(FX,FY) \otimes \mathsf{D}(FY,FZ) \arrow[r, "M"'] & \mathsf{D}(FX,FZ), & & \mathsf{D}(FX,FX).
\end{tikzcd}$$

Definition 1.41 Given two $\mathcal{V}$-endofunctors F and G between $\mathcal{V}$-categories C and D, a *$\mathcal{V}$-natural transformation α* from F to G is given by a family of maps $\alpha_X\colon I \longrightarrow \mathsf{D}(FX,GX)$ indexed over the objects of C, making the following diagram commute:

$$\begin{tikzcd}
& \mathsf{C}(X,Y) \arrow[dl, bend right, "\rho^{-1}"'] \arrow[dr, bend left, "\lambda^{-1}"] & \\
\mathsf{C}(X,Y) \otimes I \arrow[d, "F\otimes\alpha_Y"'] & & I \otimes \mathsf{C}(X,Y) \arrow[d, "\alpha_X\otimes G"] \\
\mathsf{D}(FX,FY) \otimes \mathsf{D}(FY,GY) \arrow[dr, bend right, "M"'] & & \mathsf{D}(FX,GX) \otimes \mathsf{D}(GX,GY) \arrow[dl, bend left, "M"] \\
& \mathsf{D}(FX,GY). &
\end{tikzcd}$$

Having recovered the notions of category, functor and natural transformation, it makes sense to consider an *enriched monad*, presented in the Eilenberg-Moore style.

Example 1.42 We present here some examples of enriched categories which we use later on.

1. Categories enriched over Set (with the cartesian monoidal structure) are nothing else but (small) categories in the usual sense.

2. A category enriched over Cat (which is monoidal with the cartesian structure) is called a *2-category.* The hom-sets, in this case, are categories, so they have objects (which we think of as arrows in between objects of our 2-category), and arrows, which we think of as 2-cells in between the arrows. A typical example of a 2-category is the category of (small) categories with functors as one-cells and natural transformations as 2-cells.

3. Let's take as a base for our enrichment the category consisting of the ordinal 2, which is monoidal with the meet operation as tensor product and 1 as unit. A category enriched over it consists of objects and, for each pair of objects, either a 0 or a 1 stand for the hom-set. In other words, we just have a relation on objects, and the axioms of an enriched category ensure that the relation is reflexive and transitive; i.e. a preorder. A 2-functor is an order preserving function on the objects. Therefore, the category of 2-categories is equivalent to Pre.

4. The category Pre is enriched over Set (because it is locally small), but also over itself, when we put on the hom-sets $\mathsf{Pre}(X, Y)$ the preorder defined as $f \leq g$ if and only if $f(x) \leq g(x)$ in Y for all x in X.

For a $\mathcal{V}$-category C, we should not think of an hom-set as a set of arrows. However, there is a way to get a standard category given an enriched one, by considering as arrows from X to Y in C all the $\mathcal{V}$-arrows $I \longrightarrow \mathsf{C}(X, Y)$. This allows us to associate with each $\mathcal{V}$-enriched category C a category in the usual sense, which we denote by C_o (see [35] for details).

Let $\mathcal{V}$ be an lλp symmetric monoidal closed category, in the sense of Definition 1.17, such that I is λ-presentable and the tensor of two λ-presentable objects is again finitely presentable (Kelly calls these categories *locally presentable as closed categories*, in [36]). In this case, it makes sense to talk about presentability for a $\mathcal{V}$-category. The notion of λ-filtered colimit is extended to the enriched setting, therefore it makes sense to ask for a representable functor $\mathsf{C}(X, -)$ to preserve λ-filtered colimits. As above, we shall call *λ-presentable* those objects for which this happens, and we shall say that C is *λ-presentable* if there is a small generating set of finitely presentable objects. The definition of signature extends consequently.

Also, the operation $\otimes$, extends naturally when replacing Set in (1.6) by $\mathcal{V}$ and letting X be a $\mathcal{V}$-object. This allows us to calculate left Kan extensions, and therefore to build the functor F_Σ for a given signature. More generally, the whole chain of adjunctions (1.16) still exists, and we can present an algebraic theory on such categories by means of operations and equations.

However, when moving from standard categories to enriched ones, we can not talk directly of the Eilenberg-Moore category for a monad. That is because now it makes no sense to consider a map from TX to X, given that such maps simply don't exist within the enriched context. An alternative way to approach the issue requires a couple of remarks.

First of all, let's note that, given an object X in an lλp $\mathcal{V}$-category C, we can define a functor from the functor category $[\mathsf{C}_\lambda, \mathsf{D}]$ to D by mapping the functor T to $(\mathsf{Lan}_{J_\lambda} T)X$. This functor has a right adjoint, which we denote by $\langle X, -\rangle$, so that

$$\mathsf{D}((\mathsf{Lan}_{J_\lambda} T)X, Y) \cong [\mathsf{C}_\lambda, \mathsf{D}](T, \langle X, Y\rangle).$$

A simple calculation shows that, for a λ-presentable object c in C, $\langle X, Y\rangle c = [\mathsf{C}(c, X), Y]$, where $[U, B]$ is the U-fold product of B, or, more formally, the representing object for the functor $[U, \mathsf{C}(\_, B)]\colon \mathsf{C} \longrightarrow \mathcal{V}$, thus being characterised by the isomorphism

$$\mathsf{C}(A, [U, B]) \cong \mathcal{V}(U, \mathsf{C}(A, B)). \tag{1.19}$$

When $\mathsf{C} = \mathcal{V} = \mathsf{Set}$, this is just exponentiation. If X is λ-presentable, the functor $\langle X, -\rangle$ is λ-accessible and, when $\mathsf{D} = \mathsf{C}$, $\langle X, X\rangle$ is a λ-accessible monad on C.

If $\mathcal{V}$ is Set, then one can prove that to give an algebra $TX \longrightarrow X$ for the monad T is the same as giving a monad morphism from T to $\langle X, X\rangle$. This latter notion makes sense also in the enriched setting. Furthermore, if $\mathsf{T} = \mathsf{T}_\Sigma$, then any such morphism is determined by a natural transformation from F_Σ to $V_\lambda\langle X, X\rangle$ which, if read in C_o, corresponds to a collection of maps $\Sigma c \otimes \mathsf{C}(c, X) \longrightarrow X$ for c in C_λ, thus recovering the notion of an interpretation for the operations declared by Σ, i.e. a model for the signature (see [37] for details).

Analogously, in presence of equations, the notion of an algebra for $\mathsf{T}_{\langle\Sigma,E\rangle}$ over the object X is replaced by that of a monad morphism $\alpha\colon \mathsf{T}_\Sigma \longrightarrow \langle X, X\rangle$ such that $\alpha\lambda = \alpha\rho$, with the notations of (1.18).

A survey of some of the structures which can be captured with this presentation is given in [51]; others are presented in [20, 43]. We shall propose here some examples of theories which can be expressed within this framework, hoping with this to provide both some intuition on the rather abstract and complex mathematics which we have presented, and to show that it was indeed worth the effort.

Example 1.43 The first, and definitely the easiest, example is that of the power category Set^K, where K is a set. We can think of its elements as sorted sets, so that an element x of a set X_k in a family $(X_k)_{k\in K}$, has sort k. This category is clearly enriched over Set, and it is locally presentable in the sense of Kelly, but this is not saying more than just noting that it is a locally finitely presentable category. Finitely presentable objects are K-tuples of finite sets, only finitely many of which are non empty. To give a signature in Set^K is therefore to specify, for each arity $(X_k)_{k\in K}$, a K-tuple of sets $(F_k)_{k\in K}$ so that each element in F_k represents a function taking X_h many elements of type h (for each non empty set X_h in the arity) and returning a result of type k.

We shall give here an example of such a sorted theory: that is, the category of sets with a group action on them. In general, one fixes a specific group G and considers the category of G-sets as algebraic over **Set**. In doing this, one unary operation is defined for each scalar $g \in G$. Here we want to emphasise the fact that the multiplication by scalar is a binary operation, taking two elements of different sorts as inputs. We shall therefore work in the category Set^2, and we will represent its objects as pairs (G, X), which are intended to be the carrier of a group and a set on which it acts, respectively. Our signature will have to define all the operations of the group, as well as those of the action. We therefore have the following:

group operators

$$\begin{aligned}\Sigma(\emptyset,\emptyset) &= (\{e\},\emptyset)\\ \Sigma(1,\emptyset) &= (\{(\ )^{-1}\},\emptyset)\\ \Sigma(2,\emptyset) &= (\{*\},\emptyset)\end{aligned}$$

group action

$$\Sigma(1,1) = (\emptyset,\{\cdot\})$$

Here, e stands for the neutral element of the group, whereas $*$ and $(\ )^{-1}$ are the multiplication and the inverse operation symbols, respectively. The operation $\cdot$ is in the second component of the pair because it returns an element of the set, and not of the group.

Equations are now going to impose all the structure of a group on the first component, and enforce the properties of the group action on the $\cdot$ operation. If n is thought of as a set with n elements, then we refer to them as $g_1, \ldots, g_n$ or $x_1, \ldots, x_n$ according to the component which they belong to, and the signature E and the two natural transformations λ and ρ take the following form:

group equations

$$E(1,\emptyset) = (\{invl, invr, idl, idr\}, \emptyset)$$

$$\begin{array}{rclrcl}
\lambda(invl) & = & g_1 * g_1^{-1} & \rho(invl) & = & e \\
\lambda(invr) & = & g_1^{-1} * g_1 & \rho(invr) & = & e \\
\lambda(idl) & = & e * g_1 & \rho(idl) & = & g_1 \\
\lambda(idr) & = & g_1 * e & \rho(idr) & = & g_1
\end{array}$$

$$E(3,\emptyset) = (\{ass\}, \emptyset)$$

$$\lambda(ass) = (g_1 * g_2) * g_3 \qquad \rho(ass) = g_1 * (g_2 * g_3)$$

group action

$$E(2,1) = (\emptyset, \{prod\})$$

$$\lambda(prod) = (g_1 * g_2) \cdot x_1 \qquad \rho(prod) = g_1 \cdot (g_2 \cdot x_1)$$

$$E(\emptyset,1) = (\emptyset, \{unit\})$$

$$\lambda(unit) = e \cdot x_1 \qquad \rho(unit) = x_1$$

The equation *prod* takes three arguments: two are elements of the group (the g_i's in the equation), the third is an element of the set. Notice how the arities of the equations are not related to the arities of the operations.

Example 1.44 Cat is enriched over Set, as well as over itself and over Gpd (categories where all morphisms have an inverse), and locally finitely presentable as such, the finitely presentable objects being finite colimits of finite categories. The different enrichments determine different algebras for F_Σ. Let's recall from (1.5) that

$$F_\Sigma \mathsf{C} = \int^{c \in \mathcal{N}} \Sigma c \otimes \mathsf{Cat}(c, \mathsf{C}).$$

Different enrichments result in the operation $\otimes$ acting in different ways. We know that the coend in this case is just a coproduct, and, by selecting the c-th component, an F_Σ-algebra determines a map $\Sigma c \otimes \mathsf{Cat}(c, \mathsf{C}) \longrightarrow \mathsf{C}$ in Cat. Depending on the enrichment we consider, $\mathsf{Cat}(c, \mathsf{C})$ will be a set, a groupoid, or a category. Consequently, according to the adjunction in (1.6) which defines $\otimes$, the map will correspond to a morphism $\mathsf{Cat}(c, \mathsf{C}) \longrightarrow \mathsf{Cat}(\Sigma c, \mathsf{C})$ in either Set or Gpd, or even Cat. The practical difference which this entails is that the map will preserve more or less structure of the hom-set, and this reflects also in the notion of algebra morphism. For example, enriching over Gpd allows us to define structures on a category and morphisms which preserve this structure up to isomorphism, whereas enriching over Set would enforce a strict preservation of the structure itself. See [51] for a more detailed discussion.

Let's take as an example categories with a terminal object. We want to present them as an algebraic theory over Cat. We shall therefore get a monad $\mathsf{T}_{\Sigma_\top}$ which associates with each category its extension by means of a terminal object $\top$. We already saw the signature in Example 1.16. Equations will put $\circ(X) = \top$ and $\bullet(X) = X$ for each X in the source category. These two equations will then have arity 1, the one object trivial category. Finally, we shall need an equation of arity 2 (the preorder with two elements considered as a category). This is going to ensure uniqueness of the mediating arrow to the final object; therefore it will say that, composition of any arrow from X

to Y with the mediating arrow from Y gives back the mediating arrow from X. So, for the arity $X \xrightarrow{f} Y$, the equation will say $!_Y f = !_X$ [20].

Example 1.45 In [43], Ghani and Lüth gave a very nice presentation of term rewriting systems as algebraic theories over Pre. The category of preorders and non descending maps is again enriched over Set as well as over itself. Here, we are interested in seeing it as enriched over itself (see Example 1.42-4). Let's consider the term rewriting system given by a signature S and rewrite rules collected in a set R. We want to find a signature Σ and equations on Pre such that the algebras for the corresponding monad are models of the term rewriting system $\langle S, R\rangle$. We interpret the signature as we already did in Set, and define $\Sigma(n) = S_n$, where n stands both for the discrete preorder on n elements and for the cardinal n. Our rules also have an arity: that is the total number of different variables appearing on either handside of the rule. When we write $X \vdash \rho\colon t \rightarrow s$, we suppose that all the variables in X appear in either t or s. For each such rule we define three term constructors: two will be standing for t and s, and one will stand for the rule which we are defining. Equations will then put the source and the target of the rewrite rule equal to the two new term constructors, thus forcing the relation (see [43] for details).

To make things clear, let's consider the classical example of the following TRS for addition on natural numbers: Σ has a constant 0, a unary symbol s (the successor operation) and a binary symbol $+$. The rewrite rules are $\{x\} \vdash \sigma\colon 0 + x \rightarrow x$ and $\{x, y\} \vdash \tau\colon s(x) + y \rightarrow s(x + y)$. In order to render this categorically, we define our signature Σ as follows:

$$\begin{aligned}\Sigma(\emptyset) &= \{0\}\\ \Sigma(\bullet) &= (s \quad l_1 \rightarrow^{\sigma} r_1)\\ \Sigma(\bullet \quad \bullet) &= (+ \quad l_2 \rightarrow^{\tau} r_2)\end{aligned}$$

Here we enclose in round brackets the preorder which is being defined and we label the arrows with their term constructor just for clarity. The signature is defined as the empty preorder on any other arity.

Equations will now be the following:

$$\begin{aligned}E(\bullet) &= (e_1 \quad e_2)\\ E(\bullet \quad \bullet) &= (e_3 \quad e_4)\end{aligned}$$

This is not very useful, if we do not specify their left and right handsides of the equations:

$$\begin{aligned}\lambda(e_1)_{\{x\}} &= l_1(x) & \rho(e_1)_{\{x\}} &= 0+x\\ \lambda(e_2)_{\{x\}} &= r_1(x) & \rho(e_2)_{\{x\}} &= x\\ \lambda(e_3)_{\{x,y\}} &= l_2(x,y) & \rho(e_3)_{\{x,y\}} &= s(x)+y\\ \lambda(e_4)_{\{x,y\}} &= r_2(x,y) & \rho(e_2)_{\{x,y\}} &= s(x+y).\end{aligned}$$

1.2.7 Summary

This closes the presentation of the classical theory. We have taken the classical notions of signature, equation, and model for a theory, and we have translated them into the language of category theory, thus extending them to many different categories. All we need is a locally presentable enriched category, and then we can present any algebraic theory by means of signature and equations. The examples above should provide a clear and sufficient motivation for introducing these notions.

In the remainder of the thesis, we shall focus on different possible dualisations of this theory, which give a way to reason about structures with an infinitary flavour. This will come as a result of considering not the initial algebra of an endofunctor, but rather the final coalgebra. Also, in dualising the collection of the $X + F$-algebras, we can choose to consider either the $X \times F$-coalgebras or the $X + F$-coalgebras, thus getting different structures. An analysis of these possible dualisations will be the core of the next Chapter.

Chapter 2

Dualising Algebras

In this chapter we explore different possible dualisations of the theory presented in Chapter 1. Of course, one could dualise everything straight away, by simply instantiating the theory in the case of a category C^{op} and reading the results back in C, but this is not what we really want to do. In particular, there are parts of the theory which we want to leave unchanged. A signature, for example, will still be presented in the same way. Remember, though, that in the classical theory we take the endofunctor generated by a signature Σ and we build the free monad over it, which is pointwise the carrier of the initial $X + F_\Sigma$-algebra. Here, two different and independent dualisations can be performed. On the one hand, one could consider the final $X + F_\Sigma$-coalgebra (or better, its carrier). On the other, one could consider the product, instead of the coproduct, and focus on either the initial algebra or the final coalgebra for the endofunctor $X \times F_\Sigma$.

In Table 2.1 below, we give a synoptic image of what one achieves in the different cases, in terms of the structure arising on the endofunctors.

	Monads	Comonads
Initial Algebras	$\mu Y.X + FY$	$\mu Y.X \times FY$
Final Coalgebras	$\nu Y.X + FY$	$\nu Y.X \times FY$

Table 2.1: Algebras and Coalgebras forming Monads and Comonads

The top-left corner of the table is the Kelly-Power framework which we already described. This Chapter will focus on a study of the two cases on the bottom line. In particular, in Section 2.3 we shall consider the comonad structure which is carried by the collection of the final $X \times F$-coalgebras, whereas in Section 2.4 we shall focus on the monad arising from the final $X+F$-coalgebras.

Before approaching the subject, though, it is essential to understand what coalgebras are, how they relate to signatures, and how final coalgebras model infinitary behaviours.

The study of coalgebras has long been considered less relevant to computer science and most activity concentrated on studying universal algebra using the initial algebra semantics. Recently, however, a lot of interest arose, as the dual notions were seen to model behaviour of systems, specifications of dynamic systems, to find models of concurrency, modal logic, infinite type theory, recursion theory and in many other areas [31, 52, 57, 56, 40, 13, 5, 2, 46, 11, 17].

2.1 Coalgebras and Behaviours

Coalgebras provide a different way of thinking about a signature Σ. Consider a set B of possible states of a machine, and think of Σ symbols as possible outputs. When the machine leaves a state s to reach a new state s', it produces as output the symbol corresponding to the operation which it has performed. When the symbol has arity n, s' will be an element of a set $\{s'_1, \ldots, s'_n\}$ of reachable states via that n-ary operation. The behaviour of the machine, is then described by a function from B to the disjoint union $\coprod_{n \in \mathbb{N}} \Sigma_n \times B^n$, and that is exactly $F_\Sigma B$, as we saw in (1.8). Therefore, we get a map which is dual to an algebra structure. This is what we call a *coalgebra.*

Definition 2.1 Let F be an endofunctor on a category C. The category $F-\mathsf{Coalg}$ of *F-coalgebras* has as objects pairs (A, α), where $\alpha: A \longrightarrow FA$ is a C-map. A map between two such (A, α) and (B, β), called an *F-coalgebra homomorphism,* is a C-arrow $f: A \longrightarrow B$ such that the following square commutes:

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, "\alpha"'] & B \arrow[d, "\beta"] \\
FA \arrow[r, "Ff"'] & FB.
\end{tikzcd}$$

There is a canonical *forgetful functor* $U: F-\mathsf{Coalg} \longrightarrow \mathsf{C}$, mapping a coalgebra (A, α) to its *carrier* object A in C, and mapping a coalgebra homomorphism $f: (A, \alpha) \longrightarrow (B, \beta)$ to f itself.

We are slightly abusing the notation, here, by using the same notation for several different forgetful functors, but we believe this should not cause any serious confusion.

Example 2.2 Many examples of coalgebras are described in [52]. We provide some here.

1. Let's consider the signature consisting of a constant symbol $\mathsf{0}$ for termination, and a unary symbol s which moves the system one step further. Then $F_\Sigma X$ is the set $1 + X$, where the added element is precisely $\mathsf{0}$. A coalgebra for this endofunctor is a map $\alpha\colon A \longrightarrow 1 + A$. An element $a \in A$ is sent by α to either $\mathsf{0}$, in which case the system terminates, or to $\mathsf{s}(a')$ for a new state $a' \in A$, from which it can produce another result, by applying α again. The symbol s is here quite irrelevant, because there is only one possible transition for each state, and the fact that we label it does not bring any new insight.

2. On the other hand, if we could observe different possible evolutions of the system, for example by getting different outputs from it, then it would be important to distinguish them. This is what brought many researchers to consider the notion of a *labelled transition system*. Categorically, these are just coalgebras. If O is a set of outputs, we can consider a unary function symbol for each element in O, and, together with the termination symbol $\perp$, we get a signature whose corresponding endofunctor F maps a set A to $1 + O \times A$. An F-coalgebra structure maps each state a to either $\perp$ (if the system halts in that state) or to a pair (o, a'), where o is the output and a' is the new state in which the machine ends up after performing the transition. Equivalently, we could label the transitions by the output they produce, and

say that the machine, performs a transition o from the state a to a'. This we shall sometimes write as $a \xrightarrow{o} a'$, and we shall write $a\downarrow$ when the machine halts in the state a. An instance of this is the set of finite and infinite words over an alphabet S, which we shall denote by $S^{\infty} = \{(w_1, w_2, \dots) \mid w_i \in S\}$. Given a nonempty word, we can extract the first letter out of it. This will be a symbol from S, therefore an output, and will leave us with the remainder of the word. On the empty word ε, the operation gives no result. We can describe this by means of a coalgebra $S^{\infty} \longrightarrow 1 + S \times S^{\infty}$ and we get

$$(w_1, w_2, w_3, \dots) \xrightarrow{w_1} (w_2, w_3, \dots) \qquad \varepsilon\downarrow.$$

3. We can also address nondeterminism, within this framework. If, for example, we have a machine which, from a state $s \in S$, can enter any state in a subset of S, then we can describe its behaviour by means of a map from S to the powerset $\mathcal{P}(S)$.

Although the definitions of algebra and coalgebra are dual, the categories $F-\mathsf{Alg}$ and that $F-\mathsf{Coalg}$ are not. In fact, F determines an endofunctor F^{op} on the dual of C, C^{op}. It is trivial to observe that $F^{\mathrm{op}}-\mathsf{Alg}$ is the dual category of $F-\mathsf{Coalg}$. So, it is still true that coalgebras are dual to algebras, but on different functors. Taking care of this, we can therefore translate all theorems stated for algebras in the coalgebraic framework, without any need to prove them again. For instance, Proposition 1.23 dualises at once, proving, because limits in C correspond to colimits in C^{op} and vice versa, the following.

Proposition 2.3 *The forgetful functor $U\colon F-\mathsf{Coalg} \longrightarrow \mathsf{C}$ creates all colimits existing in C, as well as those limits which are preserved by F.*

Analogously, Lambek's lemma (see page 35) dualises to give the following.

Lemma 2.4 *Whenever the final coalgebra (T, τ) for an endofunctor F on a category C exists, the map τ is an isomorphism.*

Because creation of colimits imposes preservation, we now have half the conditions for applying Freyd's special adjoint theorem [44, p.125] to deduce that U has a right adjoint. To this end, we need to show that $F-\mathsf{Coalg}$ is cocomplete and has a set of generators. This latter condition holds whenever C and F are accessible. In this case, by Corollary 2.75 in [8], we have that $F-\mathsf{Coalg}$ is accessible too, hence having a generating set of presentable objects. If C is also cocomplete (i.e. by Proposition 1.14, if it is locally presentable), then Proposition 2.3 ensures that $F-\mathsf{Coalg}$ is cocomplete too, and by Freyd's theorem we know that U has a right adjoint. We have just proved the following result.

Proposition 2.5 *Let F be an accessible endofunctor on a locally presentable category C. Then, $F-\mathsf{Coalg}$ is locally presentable and there is an adjunction*

$$\begin{tikzcd} F-\mathsf{Coalg} \arrow[r, shift left, "U"] \arrow[r, phantom, "\perp"] & \mathsf{C}. \arrow[l, shift left, "R"] \end{tikzcd} \tag{2.1}$$

Remark 2.6 It is important to notice that the result does *not* ensure that the rank of presentability is preserved when forming the category of coalgebras. More specifically, Theorem 2.72 in [8] ensures that $F-\mathsf{Coalg}$ is

μ-accessible for a regular cardinal μ which, in general, is only known to be higher than λ, even if the same λ is the rank of presentability of C and of accessibility of F. As an example, consider the covariant finite powerset functor $\mathcal{P}_{\mathrm{fi}}$. This is easily shown to be finitary on Set, which is itself lfp, but $\mathcal{P}_{\mathrm{fi}}-\mathsf{Coalg}$ is not lfp [13]. At least in Set, this seems to be an anomaly of ω, since in [7] the authors show how, for any F of rank $\lambda > \omega$, the category of F-coalgebras is locally λ-presentable and the right adjoint to U is λ-accessible.

2.2 Final Coalgebras

In the case of algebras, the universal property of initial algebras shows them as the denotational semantics for the signature. Dually, here, we focus our attention on final coalgebras, which model all possible observable behaviours of a Σ-system.

In general, final coalgebras need not exist. In fact, the powerset functor on Set cannot admit one, otherwise this would be, by Lambek's lemma, a set in bijection with its powerset (as a matter of fact, the powerset functor does not even admit an initial algebra, for the same reason; in order to find a functor which has an initial algebra but not a final coalgebra, we can look at the functor $F\colon \mathsf{FinSet} \longrightarrow \mathsf{FinSet}$ mapping an object X to $2 \times X$). Whenever C is locally presentable and F is accessible, though, the existence of a right adjoint for U ensures, since C is complete, that $F-\mathsf{Coalg}$ has a terminal object. Therefore, we have a final coalgebra. Unfortunately, Freyd's adjoint theorem is very unconstructive, therefore we do not know much about the structure of this coalgebra. Besides, these are not the only cases where we know of its existence.

2.2.1 Constructing Final Coalgebras

Since the notion of final coalgebra is dual to that of initial algebra, we can dualise the construction of the initial algebra chain (1.11), and consider the following.

Let F be an endofunctor on a category C with terminal object T. The *final F-coalgebra cochain* in C is the cochain

$$\mathcal{B} = B_0 \xleftarrow{\psi_{1,0}} B_1 \xleftarrow{\psi_{2,1}} \cdots \xleftarrow{\psi_{\mu,i}} B_\mu \xleftarrow{\psi_{j,\mu}} \cdots \tag{2.2}$$

which is inductively defined as follows. The objects will be

$$\begin{aligned} B_0 &= T \\ B_{n+1} &= F(B_n) \quad \text{for any non-limit ordinal } n \\ B_\mu &= \lim D \quad \text{for a limit ordinal } \mu \end{aligned}$$

where D is the cochain constructed until that point. Maps between them will be

$$\begin{aligned} \psi_{1,0}\colon B_1 \longrightarrow B_0 &= \text{the unique arrow to } T \\ \psi_{j,i}\colon B_j \longrightarrow B_i &= F(\psi_{i-1,j-1}) \text{ for non-limit ordinals } i < j \\ \psi_{\mu,i}\colon B_\mu \longrightarrow B_i &= \text{the } i\text{-th projection for a limit ordinal } \mu > i \\ \psi_{j,\mu}\colon B_j \longrightarrow B_\mu &= \text{the map determined by the family } (\psi_{j,l})_{l<\mu} \\ & \text{ for a limit cardinal } \mu < j \end{aligned}$$

The dual to Lemma 1.24 clearly holds, thus saying that, whenever the chain converges at the step λ, i.e. when $\psi_{\lambda+1,\lambda}\colon B_{\lambda+1} \longrightarrow B_\lambda$ is an isomorphism, then $\psi^{-1}_{\lambda+1,\lambda}$ is the structure map of a final object in $F-\mathsf{Coalg}$.

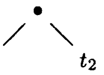
For this to hold, it is enough that F *preserves limits of ω-cochains.* In this case, taking the limit of the first ω steps of the cochain gives us the carrier of the coalgebra [13]. Such functors are often known as *(ω-)continuous* in the literature, and both the notion and the result clearly extend to any limit ordinal λ. They do not include all the accessible ones. In fact, the functor $\mathcal{P}_{\mathrm{fi}}$ which maps each set to the set of its finite subsets is finitary, but not ω-continuous [57, p.27]. A more comprehensive result was proved by James Worrell [58], who showed that whenever F preserves monos and is accessible, the coalgebra chain must converge, and in **Set** it does so in at most $\omega + \omega$ steps.

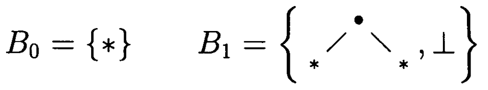
Example 2.7 The class of ω-continuous functors includes all functors arising as the left Kan extension of some finitary signature [52]. For example, if we consider the signature consisting of a constant $\mathsf{0}$ and a unary symbol s of Example 2.2–1, the corresponding functor $FX = 1 + X$ is ω-continuous, and therefore it has as a final coalgebra the limit of the first ω steps of (2.2). In this specific case, we can describe B_i as the set $\{\mathsf{0}, \dots, \mathsf{s}^{i-1}\mathsf{0}\}$, and the maps $\psi_{i+1,i}\colon B_{i+1} \longrightarrow B_i$ are defined as

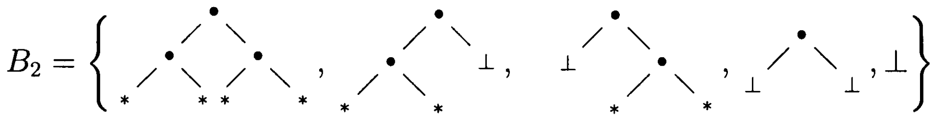
$$\psi_{i+1,i}(\mathsf{s}^j\mathsf{0}) = \begin{cases} \mathsf{s}^j\mathsf{0} & \text{if } j < i \\ \mathsf{s}^{i-1}\mathsf{0} & \text{if } j = i \end{cases}$$

The limit of the chain will then consist of the set of lists of the form $(x_0, x_1, \dots)$, where $x_i \in B_i$, such that $\psi_{i,i-1}(x_i) = x_{i-1}$. Because of the way the maps are defined, such lists must be of the form $(\mathsf{0}, \mathsf{s0}, \mathsf{s}^2\mathsf{0}, \dots)$ and they can either grow up indefinitely or stabilise at some $\mathsf{s}^n\mathsf{0}$, and constantly repeat that entry. The set of such lists is clearly in bijection with the set $\mathbb{N} \cup \{\infty\}$, and forms the carrier of the final F-coalgebra.

Given any other coalgebra $\gamma: C \longrightarrow 1 + C$ for the functor F, the mediating map $\phi: C \longrightarrow \mathbb{N} \cup \{\infty\}$ will associate with each $c \in C$ the number of iterations of γ which we can perform on c before ending up in the 1 component; that is, if $*$ is the element of 1, $\phi(c)$ will be the least natural number n such that $\gamma^n(c) = *$. If such a number does not exist, then γ iterates indefinitely on c, therefore we put $\phi(c) = \infty$.

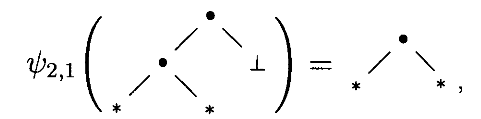
Example 2.8 Another example is that of the functor G arising from a signature consisting of one binary symbol $\bullet$ and a constant $\perp$. Formula (1.5) gives an explicit calculation for G: $GX = 1 + X^2$. If we adopt the tree notation for terms, thus writing  for $\bullet(t_1, t_2)$, the sets in the chain now take the form





and so on. We can think of the set B_n as the set of binary trees of depth at most n, where all internal nodes are labelled by $\bullet$, leaves at depth less than n are labelled by $\perp$, and leaves at depth n are labelled by $*$.

The map $\psi_{n+1,n}$ will send a tree of depth $n+1$ to its truncation at depth n, and leave a tree of lower depth unchanged. So, for example,



whereas $\psi_{2,1}(\perp) = \perp$.

An element in the limit will be a sequence $(t_0, t_1, t_2, \ldots)$ of trees such that t_n is the truncation at depth n of t_m for any $m \geq n$. We can therefore think of such sequences as successive approximations to a (possibly) infinite tree with internal nodes labelled by $\bullet$ and leaves labelled by $\perp$. If a sequence stabilises at some $n \in \mathbb{N}$ (i.e. if for all $m \geq n$ one has $t_m = t_n$), then the corresponding tree will be finite, and precisely t_n. The carrier of the final coalgebra is then the set of finite and infinite trees (with nodes labelled by $\bullet$ and leaves by $\perp$), or equivalently, the set of finite and infinite terms built over the signature.

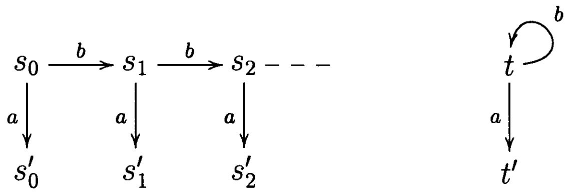
Here, the mediating map ϕ from a coalgebra $\gamma: X \longrightarrow GX$ maps a state $x \in X$ to the binary tree representing its evolution. This is built by successively instantiating the coalgebra structure. If $\gamma(x) = *$, i.e. if it falls in the 1 component of the coproduct $1 + X^2$, then $\phi(x) = \perp$; otherwise $\gamma(x) = (x_1, x_2) \in X^2$, and we map x to the tree which starts with $\bullet$ and has as a left branch the tree corresponding to x_1 and as a right branch the tree corresponding to x_2. The function ϕ is said to be defined by *corecursion.*

The argument we just showed generalises straightforwardly to any finitary signature Σ on Set, and we get the following.

Proposition 2.9 *The carrier set of the final F_Σ-coalgebra is the set of closed terms with finite and infinite depth built over the* Set*-signature Σ, and it is obtained as the limit of the final coalgebra cochain in ω steps.*

The two examples above suggest the intuition that the mediating map from any coalgebra to the final one should map any element of the carrier to the infinite tree which completely describes its observable behaviour.

The notion of *bisimulation* encodes the idea of two behaviours being indistinguishable by means of simple observation [47]. Consider for example two systems S and T:



where by saying that S is a system we mean that the s_i's are states of a machine whose transitions from one state to another are labelled by a symbol, in this case from the set $L = \{a, b\}$ (and likewise for T). Then, for any state in S there is a state in T such that any action performed by S can be performed by T and vice versa. For example, the state s_0 in S can perform an a action and move to the state s'_0, and the state t can perform an a action to t'. Likewise, s_0 goes by b to s_1, whereas t goes by b to itself. The target states, in both cases, are still related by the same property. If we consider the relation $\{(s_i, t) \mid i \geq 0\} \cup \{(s'_i, t') \mid i \geq 0\}$ between states of the two systems, this will satisfy the property that for every pair in the relation, if either of the two elements can move to another state within its own system, the other one can perform within its system a transition with the same label, and the pair consisting of the two target states will still be in the relation. This is what in concurrency is called a *bisimulation* between the two systems.

Categorically, we can model the two systems as coalgebras for the endofunctor $FX = \mathcal{P}_{\mathrm{fi}}(L \times X)$. In the first case, the carrier will be the set $S =$

$\{s_i, s'_i \mid i \geq 0\}$, with a transition map sending s_i to the set $\{(a, s'_i), (b, s_{i+1})\}$ and s'_i to the empty set; in the second, the carrier will be $T = \{t, t'\}$, with structure map which sends t to $\{(a, t'), (b, t)\}$ and t' to the empty set. The relation can then be represented as a span

$$\begin{tikzcd} S & R \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & T \end{tikzcd}$$

where R is a coalgebra and π_i is a coalgebra homomorphism. This is what we take as our general definition of bisimulation between coalgebras.

Definition 2.10 Given coalgebras (A, α) and (B, β) for a C-endofunctor F, a *bisimulation* between them is a coalgebra (R, ρ) together with a pair of coalgebra morphisms $\pi_A\colon (R, \rho) \longrightarrow (A, \alpha)$ and $\pi_B\colon (R, \rho) \longrightarrow (B, \beta)$ such that π_A and π_B are jointly monic in C.

Note that, in Set, the graph of any morphism $f\colon (A, \alpha) \longrightarrow (B, \beta)$ of coalgebras, considered as a subset of $A \times B$, is a coalgebra, and the two projections on A and B make it into a bisimulation.

A different approach to building final coalgebras [31], at least in Set, is that of considering a quotient of the collection of all the coalgebras which identifies all bisimilar states.

Following the intuition of the final coalgebra being the collection of all possible behaviours of a system, and being systems modelled by coalgebras, it is natural to build the carrier of the final coalgebra as the disjoint union of all coalgebras, letting each element represent its own behaviour. Because $F-\mathsf{Coalg}$ is cocomplete, we can define a coalgebra structure on such a union, which makes it into a coproduct. The inclusions then make it a weakly final coalgebra. Uniqueness of the mediating maps fails because a state in a coalgebra could be mapped to several bisimilar states in the coproduct. We should therefore force bisimilar states to be identified, in order to get the result.

Unfortunately, there is a big flaw in this argument, in that the coproduct of *all* coalgebras in general does not exist, for size reasons. There is a way around the problem, though, whenever we have a proper set of coalgebras $\mathcal{G} = \{G_i \mid i \in I\}$ such that for any other state a in any other coalgebra (A, α), the coalgebra $\langle a \rangle$ generated by a (i.e. the smallest subcoalgebra of (A, α) containing a) is bisimilar to G_i for some $i \in I$. Such a set is called a *set of generators*, and whenever we have one, we can get a final coalgebra by means of the described construction [52].

This second construction captures more explicitly the idea of the final coalgebra being the set of all possible behaviours expressible by means of the functor. We shall return on the idea of "collecting coalgebras" when proving Theorem 3.16, and applying it in Chapter 4.

2.2.2 Relations with Initial Algebras

When a functor $F\colon \mathsf{C} \longrightarrow \mathsf{C}$ admits both an initial algebra $\iota^{-1}\colon I \longrightarrow FI$ and a final coalgebra $\tau\colon T \longrightarrow FT$, their structure maps are isomorphisms, by Lambek's lemma. By reversing either of the two and using the universal property of the other, we determine two morphisms from I to T, which can easily be proved by diagram chasing to be the same map ϕ. We therefore have the commutative square

$$\begin{tikzcd} I \arrow[r, "\phi"] \arrow[d, "\iota^{-1}"'] & T \arrow[d, "\tau"] \\ FI \arrow[r, "F\phi"'] & FT. \end{tikzcd}$$

In some good cases, ϕ has more structure. For example, if $F = F_\Sigma$ for some Set-signature Σ, then I is the set of closed finite terms, whereas T is the set of closed finite and infinite terms, and we saw in Section 1.1.2 how the latter is the Cauchy completion of the former.

Barr [13] first observed that such a phenomenon is a consequence of a more general construction, which works for any ω-continuous finitary endofunctor, provided $F\emptyset \neq \emptyset$.

Adámek brought the subject even further, generalising this construction to each lfp category [5]. He showed that, under mild conditions on an lfp category C and a continuous endofunctor F, the final coalgebra T is such that each hom-set $\hom(B, T)$ is a Cauchy complete metric space, and the set $\hom(B, I)$, is a dense subset of it.

In Section 1.1.2, we also saw how infinite terms are an ideal completion of finite ones, according to a natural ordering induced by their structure. This result also extends to hom-sets in a locally finitely presentable category, provided again some minor assumptions are satisfied [5]. Notice that, in both cases, the functor F only needs to preserve limits of ω-cochains for the existence of the initial algebra to be automatically ensured.

2.3 Final Coalgebras and Comonads

In the two previous sections, we made ourselves familiar with the notion of coalgebra, and in particular with the final one. Now we are going to put them to use in order to dualise the Kelly-Power framework. In particular, in this section we shall focus on the bottom-right entry of Table 2.1 on page 62.

Let's recall from Chapter 1 that the carrier of the initial $X + F$-algebra for an accessible endofunctor F and an object X in C defines the functor part of the free monad over F. Dually, here we are going to show that the carrier of the final $X \times F$-coalgebra is the image of X along the *cofree comonad* on F. Unfortunately, in the process of dualising the theory, we will have to deal with a rank change, which will make everything slightly more complicated. Nevertheless, we shall still manage to reconstruct part of the adjunction (1.16) in this dual context. This will allow us to introduce notions like *cosignature*, *coequations* and *comodels*, whose computational significance will be discussed.

2.3.1 Cosignatures and their Comodels

Recall that the heart of the categorical approach to universal algebra is adjunction (1.16). The dualisation outlined in this section can be summed up as replacing the left adjoint to $U = V_\lambda(\_) \circ J_\lambda I_\lambda$ with a right adjoint and monads with comonads.

A typical situation arising in practice is to have a system and a few *methods* or *destructors* which provide a way of analysing it. These are often the only means of communication or observations of the system, and the result provided is a *partial* (typically finite) view of the whole. Categorically, we model such destructors as a functor, assigning to each arity (i.e. to each finite object) the object of methods with that arity. For this reason, although the intuition is different, the definition of a *cosignature* turns out to be formally the same as that of signature.

Definition 2.11 Let C be an lλp-category with arities $\mathcal{N}_\lambda$. A *λ-cosignature* is a functor $B\colon \mathcal{N}_\lambda \longrightarrow \mathsf{C}$.

Recall that, before, we constructed a λ-accessible endofunctor from a signature by first taking a left Kan extension, and then using equivalence (1.7) to get a λ-accessible endofunctor. By duality, here we take the right Kan extension of a cosignature B to obtain a functor $\mathsf{Ran}_{I_\lambda}B\colon \mathsf{C}_\lambda \longrightarrow \mathsf{C}$. The standard formula for the right Kan extension gives us

$$(\mathsf{Ran}_{I_\lambda}B)X = \prod_{c\in\mathcal{N}_\lambda}[\mathsf{C}(X,c),Bc] \tag{2.3}$$

where the operation $[-,-]$ is defined as in (1.19).

Thus, although signatures and cosignatures are formally the same, the endofunctors they generate are very different. For example, note that, while the default value for signatures is 0, if there is a single arity c such that $B(c) = 0$, then $(\mathsf{Ran}_{I_\lambda}B)(X) = 0$. In fact, the default value for cosignatures is the final object 1 since $[U,1] = 1$ and hence if c is an arity such that $B(c) = 1$, then this arity will contribute nothing to the right Kan extension. Here are two examples of Set-cosignatures which we shall explore further below.

Example 2.12 Define the cosignature B_2 by $B_2(2) = 2$ and $B_2(c) = 1$ for all other arities. Then, for a finite set X, $\mathsf{Ran}_{I_\lambda} B_2(X) = [\mathsf{Set}(X, 2), 2] = [[X, 2], 2]$, where $[A, B]$ stands for the set of functions from A to B.

Define the cosignature $B_\omega(2) = \omega$ and $B_\omega(c) = 1$ for all other arities. Then, again for a finite X, (2.3) shows that $(\mathsf{Ran}_{I_\lambda} B_\omega)(X) = [\mathsf{Set}(X, 2), \omega] = [[X, 2], \omega]$.

The reader might now expect a right Kan extension of $\mathsf{Ran}_{I_\lambda} B$ along J_λ, by duality to the classical construction. Instead, we consider a left Kan extension. The reason for this is that we consider the categories $[\mathsf{C}_\lambda, \mathsf{C}]$ and $[\mathsf{C}, \mathsf{C}]_\lambda$ as equivalent under (1.7), which is computed exactly as the left Kan extension along J_λ. The fact that we are not taking a completely dual construction, here, is of course going to generate some complications in the following, but we accept this for the reasons we just explained.

We then get a finitary endofunctor corresponding to B; that is,

$$G_B = \mathsf{Lan}_{J_\lambda} \mathsf{Ran}_{I_\lambda} B \colon \mathsf{C} \longrightarrow \mathsf{C}. \tag{2.4}$$

For example, in the case of the signature B_2 presented in Example 2.12, it is easy to compute that the corresponding functor G_{B_2} is the functor ${\mathcal{P}_{\mathrm{fi}}}^{\mathrm{c}} \circ {\mathcal{P}_{\mathrm{fi}}}^{\mathrm{c}}$, where ${\mathcal{P}_{\mathrm{fi}}}^{\mathrm{c}}$ is the contravariant finite powerset functor.

Let's now focus on the category of G_B-coalgebras. An object in G_B–Coalg is a C-arrow

$$X \longrightarrow (\mathsf{Lan}_{J_\lambda} \mathsf{Ran}_{I_\lambda} B) X.$$

If X is λ-presentable, then $(\mathsf{Lan}_{J_\lambda} \mathsf{Ran}_{I_\lambda} B) X = \mathsf{Ran}_{I_\lambda} B X$; hence, the coalgebra is indeed a map

$$X \longrightarrow \mathsf{Ran}_{I_\lambda} BX.$$

Now, using (2.3), it is easy to show the following chain of isomorphisms:

$$\begin{aligned} \mathsf{C}(X, (\mathsf{Ran}_{I_\lambda} B)(X)) &\cong \mathsf{C}(X, \prod_{c \in \mathcal{N}_\lambda} [\mathsf{C}(X, c), Bc]) \\ &\cong \prod_{c \in \mathcal{N}_\lambda} \mathsf{C}(X, [\mathsf{C}(X, c), Bc]) \qquad (2.5) \\ &\cong \prod_{c \in \mathcal{N}_\lambda} [\mathsf{C}(X, c), \mathsf{C}(X, Bc)] \end{aligned}$$

So, to every coalgebra over a λ-presentable object X corresponds a family of maps

$$(\mathsf{C}(X, c), \mathsf{C}(X, Bc))_{c \in \mathsf{C}_\lambda}.$$

These provide a *cointerpretation* of the function symbols in the cosignature.

Analogously to the algebraic case, where we defined the category of F_Σ-algebras to be the category of models for the theory, relying on the fact that from any F_Σ-algebra we could retrieve an interpretation for each term constructor, here we shall consider the category $G_B - \mathsf{Coalg}$ as the category of *comodels* for the cosignature B.

Example 2.13 A comodel on a finite set X for the cosignature B_2 is given by a coalgebra $f \colon X \longrightarrow [[X, 2], 2]$. We can interpret it as a map saying, for each state $x \in X$, which properties (i.e. subsets of X) ensure that the system will evolve to it. Via the equivalence described in (2.5), this becomes a map

$$\widehat{f} \colon [X, 2] \longrightarrow [X, 2],$$

mapping a property of X to the property entailed by it as the system evolves. This is therefore a predicate transformer.

A coalgebra morphism from a predicate transformer $\alpha\colon X \longrightarrow [[X,2],2]$ to $\beta\colon Y \longrightarrow [[Y,2],2]$ is a function $f\colon X \longrightarrow Y$ such that

$$\begin{tikzcd} X \arrow[r, "\alpha"] \arrow[d, "f"'] & {[[X,2],2]} \arrow[d, "\mathcal{P}_{\mathrm{fi}}{}^{c}\mathcal{P}_{\mathrm{fi}}{}^{c}f"] \\ Y \arrow[r, "\beta"'] & {[[Y,2],2].} \end{tikzcd} \tag{2.6}$$

Commutativity of (2.6) can be read as saying that, given a state $x \in X$, the properties of Y which entail an evolution of the system Y into $f(x)$ determine, via counterimage along f, properties of X which entail an evolution to x.

Under the adjunction, (2.6) becomes

$$\begin{tikzcd} {[X,2]} \arrow[r, "\widehat{\alpha}"] & {[X,2]} \\ {[Y,2]} \arrow[u, "-\circ f"] \arrow[r, "\widehat{\beta}"'] & {[Y,2].} \arrow[u, "-\circ f"'] \end{tikzcd}$$

The Y predicate following a predicate P is backtracked along f to a predicate on X which follows from $f^{-1}(P)$.

2.3.2 The Representing Comonad of a Cosignature

Recall that, in the algebraic case, one starts with a signature, gets the corresponding endofunctor, and then considers the free monad over it. In (2.4) we built the endofunctor in a dual way. Now it is time to consider the *cofree comonad* over it.

The notions of comonad, and in particular of cofree comonad, are dual to those of monad and free monad. More specifically, we dualise the definition in the Eilenberg-Moore style.

Definition 2.14 A *comonad* L on a category C consists of an endofunctor L on C together with two natural transformations $\varepsilon\colon L \longrightarrow \mathsf{Id}$ and $\nu\colon L \longrightarrow L^2$ making the dual diagrams to (1.13) commute. We therefore have:

$$\varepsilon_L \nu = \mathsf{id}_L = L(\varepsilon)\nu \qquad\qquad \nu_L \nu = L(\nu)\nu.$$

A *comonad morphism* between comonads (L, ε, ν) and (L', ε', ν') is a natural transformation $\phi\colon L \longrightarrow L'$ such that $\varepsilon'\phi = \varepsilon$ and $\nu'\phi = \phi^2\nu$.

Comonads and comonad morphisms form a category, which we shall denote by $\mathsf{Com}(\mathsf{C})$. This has a natural forgetful functor V to the category of C-endofunctors $\mathsf{End}(\mathsf{C})$.

With an abuse of notation, we shall denote by V also its restriction to a functor $V\colon \mathsf{ACom}(\mathsf{C}) \longrightarrow \mathsf{AEnd}(\mathsf{C})$ from the category of *accessible comonads* (i.e. those comonads whose underlying functor is accessible) to that of accessible endofunctors. Given a regular cardinal λ, we can further restrict V to a functor $V_\lambda\colon \mathsf{Com}(\mathsf{C})_\lambda \longrightarrow \mathsf{End}(\mathsf{C})_\lambda$.

The notion of cofree comonad is also dual to that of free monad.

Definition 2.15 Given an endofunctor F on C, the *cofree comonad* over it is a comonad $\mathsf{R}_F = (R_F, \varepsilon, \nu)$ together with a natural transformation $\iota\colon R_F \longrightarrow F$ such that, for any other comonad $\mathsf{L} = (L, \varepsilon', \nu')$ and any natural transformation $\phi\colon L \longrightarrow F$, there is a comonad morphism $\widehat{\phi}\colon \mathsf{R}_F \longrightarrow \mathsf{L}$ such that $\phi\widehat{\phi} = \iota$.

Algebras for a monad also have their dual version, in the notion of *coalgebra for a comonad* L. The definition obviously mirrors Definition 1.33.

Definition 2.16 A *coalgebra* for a comonad $\mathsf{L} = (L, \varepsilon, \nu)$ is an L-coalgebra $\phi: X \longrightarrow LX$ satisfying the equations expressing commutativity of the following diagrams:

$$\begin{tikzcd}
X \arrow[r, "\phi"] \arrow[dr, "\mathsf{id}"'] & L(X) \arrow[d, "\varepsilon_X"] \\
 & X
\end{tikzcd}
\qquad
\begin{tikzcd}
X \arrow[r, "\phi"] \arrow[d, "\phi"'] & L(X) \arrow[d, "L(\phi)"] \\
L(X) \arrow[r, "\nu"'] & L^2(X).
\end{tikzcd}
\tag{2.7}$$

The full subcategory of $L-\mathsf{Coalg}$ based on such coalgebras is called the category of *coalgebras for the comonad* L and it is denoted by $\mathsf{L}-\mathsf{Coalg}$.

The following result from [33] provides the setting for this discussion, and it is to be compared to Proposition 1.36.

Lemma 2.17 *The following conditions on a functor $F: \mathsf{C} \longrightarrow \mathsf{C}$ are equivalent:*

1. *(If C has products) For every object X, the functor $X \times F$ has a final coalgebra.*

2. *The forgetful functor $F-\mathsf{Coalg} \longrightarrow \mathsf{C}$ is comonadic.*

3. *The forgetful functor $F-\mathsf{Coalg} \longrightarrow \mathsf{C}$ has a right adjoint.*

4. *There is a cofree comonad on F.*

Similarly to the monadic case, when C is lλp and F is λ-accessible, Proposition 2.5 and Lemma 2.17 ensure the existence of a cofree comonad over F; whence, the following result.

Proposition 2.18 *The forgetful functor* $V\colon \mathsf{ACom(C)} \longrightarrow \mathsf{AEnd(C)}$ *has a right adjoint* R*.*

Proof. Given an accessible endofunctor F on C, the natural transformation $\iota\colon R_F \longrightarrow F$ from the functor part of the cofree comonad (R_F, ε, ν) over F to F itself, is by definition a universal arrow from V to $\mathsf{AEnd(C)}$. The existence of a cofree comonad for any accessible endofunctor therefore ensures the existence of a right adjoint to V. □

As opposed to the monadic case, the cofree comonad R_M on an endofunctor $M\colon \mathsf{C} \longrightarrow \mathsf{C}$ of rank λ need not have rank λ. As a simple counterexample, consider the endofunctor $M\colon \mathsf{Set} \longrightarrow \mathsf{Set}$ defined as $M = A \times -$ for a fixed set A, which is clearly finitary. We know that the value of R_M for a set X is given by the carrier of the final $X \times M$-coalgebra. $X \times M$ maps a set Y to $X \times A \times Y$, and its final coalgebra is easily proved to be the set of all infinite lists of pairs from $X \times A$. Now consider a countably infinite set X. $R_M X$ contains a list with infinitely many different elements from X, and this can not be an element of $R_M X_0$ for any finite subset X_0 of X, which shows that R_M has a rank larger than ω. Generally speaking, as we saw in Remark 2.6, calculating coalgebras of finitary endofunctors invariably seems to increase their rank.

Using the equivalence between $[\mathsf{C}_\lambda, \mathsf{C}]$ and $[\mathsf{C}, \mathsf{C}]_\lambda$, we now have the following functors:

$$\begin{tikzcd}
{[\mathcal{N}_\lambda, \mathsf{C}]} \arrow[r, shift left, "\mathsf{Ran}_{I_\lambda}"] \arrow[r, phantom, "\top"] & {[\mathsf{C}_\lambda, \mathsf{C}]} \arrow[l, shift left, "-\circ I_\lambda"] \arrow[r, shift left, "\mathsf{Lan}_{J_\lambda}"] \arrow[r, phantom, "\cong"] & {[\mathsf{C}, \mathsf{C}]_\lambda} \arrow[l, shift left, "-\circ J_\lambda"] \arrow[d, hook] & \mathsf{Com}(\mathsf{C})_\lambda \arrow[l, "V_\lambda"'] \arrow[d, hook] \\
 & & \mathsf{AEnd}(\mathsf{C}) \arrow[r, shift left, "R"] \arrow[r, phantom, "\top"] & \mathsf{ACom}(\mathsf{C}). \arrow[l, shift left, "V"]
\end{tikzcd} \tag{2.8}$$

Notice that we shall ignore the inclusion functors which discard the rank of a functor or a monad.

We now further define the composite functors

$$\begin{array}{ll}
W_\lambda \colon \mathsf{Com}(\mathsf{C})_\lambda \longrightarrow [\mathcal{N}_\lambda, \mathsf{C}] & \qquad W_\lambda = (V_\lambda(-)) \circ J_\lambda I_\lambda \\
R_\lambda \colon [\mathcal{N}_\lambda, \mathsf{C}] \longrightarrow \mathsf{ACom}(\mathsf{C}) & \qquad R_\lambda = R\mathsf{Lan}_{J_\lambda}\mathsf{Ran}_{I_\lambda}
\end{array}$$

As we have seen, the rank of $R_\lambda B$ may be greater than the rank of B, hence the codomain of R_λ is not the domain of V_λ, but rather $\mathsf{ACom}(\mathsf{C})$. Our partial recovery of the chain of adjunctions (1.16) consists of the following correspondence:

Lemma 2.19 *For any λ-accessible comonad $\mathsf{L} = (L, \varepsilon, \nu)$ and λ-cosignature $B \colon \mathcal{N}_\lambda \longrightarrow \mathsf{C}$, there is an isomorphism*

$$[\mathcal{N}_\lambda, \mathsf{C}](W_\lambda \mathsf{L}, B) \cong \mathsf{ACom}(\mathsf{C})(\mathsf{L}, R_\lambda B). \tag{2.9}$$

Proof. The isomorphism is shown by the following chain of natural isomorphisms provided by the two adjunctions in (2.8) and the full and faithful embedding of $[\mathsf{C},\mathsf{C}]_\lambda$ into $\mathsf{AEnd}(\mathsf{C})$:

$$\begin{aligned}
[\mathcal{N}_\lambda,\mathsf{C}](W_\lambda\mathsf{L},B) &\cong [\mathcal{N}_\lambda,\mathsf{C}](V_\lambda(\mathsf{L})J_\lambda I_\lambda,B)\\
&\cong [\mathsf{C}_\lambda,\mathsf{C}](V_\lambda(\mathsf{L})J_\lambda,\mathsf{Ran}_{I_\lambda}B)\\
&\cong \mathsf{AEnd}(\mathsf{C})(V_\lambda(\mathsf{L}),\mathsf{Lan}_{J_\lambda}\mathsf{Ran}_{I_\lambda}B)\\
&\cong \mathsf{ACom}(\mathsf{C})(V(\mathsf{L}),R\mathsf{Lan}_{J_\lambda}\mathsf{Ran}_{I_\lambda}B)\\
&\cong \mathsf{ACom}(\mathsf{C})(\mathsf{L},R_\lambda B)
\end{aligned}$$

□

So, given a λ-cosignature B, we have constructed its representing comonad $\mathsf{L_B} = R_\lambda B$, whose functor part is denoted L_B. Note that, by Lemma 2.17, the category of coalgebras for the comonad $\mathsf{L_B}-\mathsf{Coalg}$ is isomorphic to the category $\mathsf{Lan}_{J_\lambda}\mathsf{Ran}_{I_\lambda}B-\mathsf{Coalg}$. Restricting ourselves to λ-presentable coalgebras, we have that the λ-presentable coalgebras of the representing comonad $R_\lambda B$ are isomorphic to the λ-presentable models of the cosignature B as seen in (2.5). This is our partial dualisation of the result stating that the models of a signature are isomorphic to the algebras for the representing monad.

2.3.3 Coequational Presentations and their Representing Comonads

In this section, we perform the last part of the dualisation by defining *coequational presentations*, deriving a representing comonad for one such and relating its coalgebras to the models of the presentation. As we have seen in Chapter 1, equations are interpreted as a pair of monad morphisms between free monads, and the representing monad for an equational presentation is then defined to be the coequaliser of these monad morphisms. Dualising this requires a coequational presentation to form a pair of comonad morphisms between cofree comonads and taking the representing comonad for the coequational presentation to be the equaliser of these comonad morphisms. Of course, because the cofree comonad on a λ-cosignature could have rank different from λ, we need to choose the forgetful functors accordingly. This is another example of how things go wrong because of the change of rank when forming final coalgebras.

Definition 2.20 A *coequational presentation* is given by two cosignatures $B: \mathcal{N}_\lambda \longrightarrow \mathsf{C}$ and $E: \mathcal{N}_\kappa \longrightarrow \mathsf{C}$ (where the functor $R_\lambda B$ is κ-accessible), and two comonad morphisms $\sigma, \tau: R_\lambda B \longrightarrow R_\kappa E$ in $\mathsf{ACom}(\mathsf{C})$.

Under (2.9), the maps $\sigma, \tau: R_\lambda B \longrightarrow R_\kappa E$ are determined by two natural transformations $\sigma', \tau': W_\kappa R_\lambda B \longrightarrow E$ in $[\mathcal{N}_\kappa, \mathsf{C}]$, which in turn consist of families $\sigma'_c, \tau'_c: R_\lambda Bc \longrightarrow Ec$ of maps for $c \in \mathcal{N}_\kappa$.

As mentioned above, given a coequational presentation, our intention is to define its representing comonad to be the equaliser of the comonad morphisms:

$$\begin{tikzcd} G \arrow[r] & R_\lambda B \arrow[r, shift left, "\sigma"] \arrow[r, shift right, "\tau"'] & R_\kappa E \end{tikzcd} \tag{2.10}$$

Existence of such equalisers is ensured by the following result, which needs a preliminary definition, which we take from Barr [14].

Definition 2.21 A *contractible equaliser* in a category C is a pair of parallel morphisms $d_0, d_1: X \longrightarrow Y$ such that there exist maps d, s and t as in the following diagram

$$\begin{tikzcd} A \arrow[r, shift left, "d"] & X \arrow[l, shift left, "s"] \arrow[r, bend left, "d_0"] \arrow[r, bend right, "d_1"'] & Y \arrow[l, "t" description] \end{tikzcd} \tag{2.11}$$

such that

$$td_0 = \mathsf{id}_X; \qquad sd = \mathsf{id}_A; \qquad ds = td_1; \qquad d_0 d = d_1 d.$$

If $U\colon \mathsf{B} \longrightarrow \mathsf{C}$ is a functor, then a *U-contractible equaliser pair* is a parallel pair $d_0, d_1\colon X \longrightarrow Y$ in B such that its image under U is a contractible equaliser.

Note that, by the equalities described in the definition, the map d is always an equaliser in C of the pair Ud_0, Ud_1. Moreover, because all the equalities are formulated by means of compositions and identities, such equalisers are preserved by any functor, i.e. they are *absolute.*

Proposition 2.22 *The category* $\mathsf{ACom}(\mathsf{C})$ *of accessible comonads over a locally presentable category* C *has equalisers within the category of* C*-comonads.*

Proof. The statement means, in detail, that, for any parallel pair in $\mathsf{ACom}(\mathsf{C})$, there is an equaliser of it within the category of all comonads on C, and furthermore, its functor part is accessible. The proof of the result partially dualises Kelly's construction [34], where the notion of an algebraic colimit of monads (i.e. the dual notion to the one we are dealing with) is introduced and discussed thoroughly. For our purpose, it is sufficient to note the following. Let $\mathsf{L} = (L, \varepsilon, \nu)$ and $\mathsf{L}' = (L', \varepsilon', \nu')$ be two comonads, and σ, τ be two comonad morphisms from L to L'. They induce two functors $\sigma^*, \tau^*\colon \mathsf{L}-\mathsf{Coalg} \longrightarrow \mathsf{L}'-\mathsf{Coalg}$. Let's consider their equaliser in Cat, i.e. the full subcategory E of $\mathsf{L-Coalg}$ whose objects are those L-coalgebras $\gamma: X \longrightarrow LX$ for which $\sigma_X\gamma = \tau_X\gamma$. The forgetful functor $U: \mathsf{L-Coalg} \longrightarrow \mathsf{C}$ restricts to a functor U' on E, and if we prove U' to have a right adjoint such that the category of coalgebras for the comonad corresponding to the adjunction is isomorphic to E, then that comonad is precisely the equaliser of σ and τ [34].

First of all, let's present E as an *equifier*, thus making sure that it is an accessible category [8, p. 122-ff]. We achieve this by noting that the structure map of an L-coalgebra can be presented as the component of a natural transformation $\phi: U \longrightarrow LU: \mathsf{L-Coalg} \longrightarrow \mathsf{C}$, where U is the forgetful functor. Using this, we can express the fact that an algebra equates the σ and τ by asking for equality of the pair

$$\sigma\phi,\ \tau\phi: U \longrightarrow L'U$$

Note that, since σ and τ are comonad morphisms, we have no need to ask for their action to map L-coalgebras to L'-coalgebras, since this is ensured. Lemma 2.76 in [8] now ensures that, because $\mathsf{L-Coalg}$ is accessible, E is too. It is also very simple to show that U' creates colimits, since U does. Therefore, E is cocomplete, and hence locally presentable.

In particular, this implies that E is co-wellpowered and has a generating set. This, together with the fact that U' preserves colimits (since it creates them), ensures, by Freyd's special adjoint functor theorem [44, Corollary on p. 130], the existence of a right adjoint to U'.

We now only need to prove that U' is cotripleable (the dual notion to that of tripleability in the sense of Barr), and this we shall get by Beck's tripleability

Theorem, once we prove that U' reflects isomorphisms and E has, and U' preserves, equalisers of reflexive U-contractible equaliser pairs [14].

The fact that U' reflects isomorphisms is a trivial observation. Let now $d_0, d_1 \colon (X, \gamma) \longrightarrow (Y, \delta)$ be a parallel pair of L-coalgebra morphisms for which there are maps d, s and t as in (2.11) making it into a U-contractible equaliser in C.

Then, because all functors, and in particular L, preserve contractible equalisers, Ld is an equaliser of Ld_0 and Ld_1. Therefore, because $L(d_0)\gamma d = \delta d_0 d = \delta d_1 d = L(d_1)\gamma d$, there is a map $\alpha \colon A \longrightarrow LA$ such that $L(d)\alpha = \gamma d$, i.e. d is an L-coalgebra morphism. We shall show that α is an object in E, i.e. an L-coalgebra map equalising σ and τ, and this will show existence of U'-contractible equalisers. Preservation is then trivial.

Satisfaction of the unit law can be proved by chasing the following diagram, where $\varepsilon_X \gamma = \mathsf{id}_X$ because γ is an L-coalgebra:

$$\begin{tikzcd}
A \arrow[r, "\alpha"] \arrow[d, "d"'] & LA \arrow[r, "\varepsilon_A"] \arrow[d, "Ld"'] & A \arrow[d, shift right, "d"'] \\
X \arrow[r, "\gamma"'] & LX \arrow[r, "\varepsilon_X"'] & X. \arrow[u, shift right, "s"']
\end{tikzcd}$$

Analogously, for the multiplication law one has that

$$\begin{aligned}
L^2(s)L^2(d)L(\alpha)\alpha &= L^2(s)L(\gamma)L(d)\alpha \\
&= L^2(s)L(\gamma)\gamma d \\
&= L^2(s)\nu_X\gamma d \\
&= L^2(s)\nu_X L(d)\alpha \\
&= L^2(s)L^2(d)\nu_A\alpha,
\end{aligned}$$

from which it follows, because $sd = \mathsf{id}_A$, that $L(\alpha)\alpha = \nu_A\alpha$.

Finally, we need to show that $\sigma_A\alpha = \tau_A\alpha$, but this, again, follows by chasing the diagram below, where $\tau_X\gamma = \sigma_X\gamma$:

$$\begin{tikzcd}
A \arrow[r, "\alpha"] \arrow[d, "d"'] & LA \arrow[r, shift left, "\sigma_A"] \arrow[r, shift right, "\tau_A"'] \arrow[d, "Ld"'] & L'A \arrow[d, shift right, "L'd"'] \\
X \arrow[r, "\gamma"'] & LX \arrow[r, shift left, "\sigma_X"] \arrow[r, shift right, "\tau_X"'] & L'X. \arrow[u, shift right, "L's"']
\end{tikzcd}$$

□

As we did in the monadic case, where we defined algebras for the monad corresponding to a theory as monad morphisms, here we can model coalgebras for a comonad G representing the coequational presentation (B, E) by means of morphisms of comonads. First, observe that an object X in C is specified by a map $\mathsf{K}_X\colon 1 \longrightarrow \mathsf{C}$ (where 1 is the one-object category). Further, the functor category $[1, \mathsf{C}]$ is isomorphic to C, and, for any endofunctor L on C, we have

$$\mathsf{C}(X, LX) \cong [1, \mathsf{C}](\mathsf{K}_X, L \circ \mathsf{K}_X) \cong [\mathsf{C}, \mathsf{C}](\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X, L), \tag{2.12}$$

so giving an L-coalgebra $X \longrightarrow LX$ is the same as giving a natural transformation $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \Longrightarrow L$. In fact, we can prove more.

Lemma 2.23 *$\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X$ is a comonad. If X is λ-presentable, then $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X$ is λ-accessible.*

Proof. Using the standard formula for left Kan extensions, $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X(A) = \mathsf{C}(X, A) \otimes X$. If X is λ-presentable, then $\mathsf{C}(X, -)$ preserves all λ-filtered colimits, and so does $- \otimes X$ too; hence $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X$ is λ-accessible.

To have a comonad structure, we need natural transformations $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \Longrightarrow 1$ and $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \Longrightarrow \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \circ \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X$ which satisfy the comonad laws. The first of these is given by the image of the identity transformation on X under the isomorphism $[1,\mathsf{C}](\mathsf{K}_X,\mathsf{K}_X) \cong [\mathsf{C},\mathsf{C}](\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X,\mathsf{Id}_{\mathsf{C}})$. The second is given by the image under the isomorphism $[1,\mathsf{C}](\mathsf{K}_X,\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \circ \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \circ \mathsf{K}_X) \cong [\mathsf{C},\mathsf{C}](\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X,\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \circ \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X)$ of the transformation

$$\mathsf{K}_X \xRightarrow{\epsilon} \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \circ \mathsf{K}_X \xRightarrow{\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X\epsilon} \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \circ \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \circ \mathsf{K}_X$$

where ϵ is the canonical transformation $\mathsf{K}_X \Longrightarrow \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \circ \mathsf{K}_X$. That the counit and comultiplication obey the comonad laws is easily verified. □

As we saw in Theorem 1.12, every object in C is presentable, therefore $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X$ is an accessible comonad for any X, and we can strengthen equation (2.12) to obtain the promised characterisation of the coalgebras of a comonad.

Proposition 2.24 *A coalgebra for a comonad* G *is given by a* C*-object* X *and a map* $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \Rightarrow \mathsf{G}$ *in* $\mathsf{ACom}(\mathsf{C})$.

Proof. We have already seen that the structure map of the coalgebra is precisely a natural transformation between the two functors. It is then routine to verify that the properties of the structure map of a coalgebra correspond to the laws of a comonad morphism. □

If G is the equaliser of $\sigma,\tau\colon R_\lambda B \longrightarrow R_\kappa E$, then a coalgebra (X,α') determines a monad morphism $\mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \longrightarrow \mathsf{G}$, which in turn determines a monad morphism $\alpha\colon \mathsf{Lan}_{\mathsf{K}_X}\mathsf{K}_X \longrightarrow R_\lambda B$ which equalises σ and τ. When X is κ-presentable, this corresponds by (2.5) to families of equal maps $\sigma''_c\alpha''_c, \tau''_c\alpha''_c\colon \mathsf{C}(X,c) \longrightarrow \mathsf{C}(X,Ec)$ for c in $\mathcal{N}_\kappa$. With this, we have finished our partial dualisation of the classical work by Kelly and Power. Notice in particular that, in this case, we do not have an analogy of the representation result of Theorem 1.38, that is, we can not say that every accessible comonad is the equaliser of two free ones. In fact, as we said above, we don't even have a dual of adjunction (1.16), which was the main ingredient in the proof for the monadic case.

2.3.4 Relations with Other Work

Lack of examples makes it difficult to study the cases where the result does not hold, and to develop a good intuition on the subject, but we feel this approach, which was proposed in [27] can give new insights, useful in addressing co-Birkhoff theorems or the theory of coalgebraic specification [11, 17].

For example, Cirstea [17] defines an *abstract cosignature* as a functor $F\colon \mathsf{C} \longrightarrow \mathsf{C}$, where C has all finite limits and limits of ω-cochains, and F is continuous and preserves pullbacks. Then, an *observer* is given by a pair (K,c) consisting of a functor $K\colon \mathsf{C} \longrightarrow \mathsf{C}$ together with a natural transformation $c\colon U \longrightarrow KU$, where U is the forgetful functor from $F-\mathsf{Coalg}$ to Set. Finally, a *coequation* is given by two observers (K,l) and (K,r), and the author writes $\mathsf{Coalg}(\mathsf{C},F,E)$ for the full category on those F-coalgebras which satisfy a set E of equations. The natural transformation defining an observer is easily seen to determine a functor from $F-\mathsf{Coalg}$ to $K-\mathsf{Coalg}$.

Therefore, an equation consists of a parallel pair of functors from $F-\mathsf{Coalg}$ to $K-\mathsf{Coalg}$. If C is lfp and F and K arise from finitary signatures, then we know that $F-\mathsf{Coalg}$ is isomorphic to the category of coalgebras for the cofree comonad on F, and analogously for $K-\mathsf{Coalg}$. In this case, our notion of coequation agrees with the one presented in her paper, and $\mathsf{Coalg}(\mathsf{C}, F, E)$ is the category of coalgebras for the comonad representing the coequations. Cirstea insists on some finite completeness properties of the category and preservation properties of the functors, in order to make sure that she can talk about covarieties and such. In our context, we rather focus on a more abstract picture, and we try to relate transformations of comodels to cosignature morphisms, hence the different requirements on the base category.

2.4 Final Coalgebras and Monads

We now turn to the bottom-left cell of Table 2.1. We know from Proposition 2.9 that the carrier of the final coalgebra for a functor F_Σ arising as the left Kan extension of a signature is the set of finite and infinite closed terms built over the signature. We are going to show a simple characterisation of the set of finite and infinite terms over the signature but with variables from a set. On a syntactic level, it is very well known that infinite terms are closed under substitution. They are used to model infinite computations, thus providing a semantics for programs. Given a language, we can interpret any code as an infinite tree, by unfolding the recursive processes which take place therein. The fact that we can solve any (guarded) system of equations on terms ensures soundness of this semantics.

We saw in Section 1.1.3 how the solution of recursive systems of equations can be obtained by Banach's fixpoint theorem, using the fact that infinite terms form a Cauchy-complete metric space. In Section 2.2.2 we also saw how this metric is determined by the construction of the final coalgebra. It is therefore natural to expect the universal properties of final coalgebras to determine the existence of a solution to recursive systems,without necessarily going through the metric argument. In fact, given such a system, we can associate with it a coalgebra, and the mediating morphism to the final coalgebra determines the solution of the system itself.

The fact that terms built over variables are closed under substitution also has a categorical counterpart. This is precisely the same as saying that the collection of the carriers of final $X + F$-coalgebras, when X ranges over C, defines a monad.

The purpose of this section is to show precisely this last result. We shall present here our original proof of it, which works for any locally finitely presentable category and any functor arising from a signature Σ. This setting is, in fact, rather restrictive, in that existence of a final $X + F$-coalgebra for every X ensures that their collection forms a monad, as L. Moss in [48] and J. Adámek in [1] proved independently with each other and with us. We shall recall their main results here.

2.4.1 The Monad of Infinite Terms

So, let's consider a finitary signature Σ on C. In order to keep the notation simple, we shall denote by F (instead of F_Σ) the left Kan extension of Σ along

$JI: \mathcal{N} \longrightarrow \mathsf{C}$. As before, we shall write $X+F$ for the functor K_X+F and $\mathsf{T} = (T, \eta, \mu)$ for the free monad over F, for which we know, by Proposition 1.36, that the set TX is the carrier of the initial $X+F$-algebra, with structure map $\iota_X = [\eta_X, \chi_X]: X + FTX \longrightarrow TX$. We shall also write T^ν for the functor which maps an object X to the carrier of the final $X + F$-coalgebra, which has as a structure map the function $\tau_X: T^\nu X \longrightarrow X + FT^\nu X$. Note that, by Lambek's lemma, τ_X has an inverse, which we shall write $[\eta^\nu_X, \alpha_X]: X + FT^\nu X \longrightarrow T^\nu X$.

From now on, we shall restrict ourselves only to the category Set, in order to maintain readable notations. In this case, as we saw in Section 2.2.2, the unique algebra/coalgebra homomorphism $\phi: TX \longrightarrow T^\nu X$ realises $T^\nu X$ as the Cauchy completion of the metric space TX. The argument we shall present extends smoothly to categories other than Set, once we notice [5] that the hom-sets $\hom(X, T^\nu Y)$ are in this case the Cauchy completion of $\hom(X, TY)$, under very mild assumptions on C and F. We aim to show that T^ν is the functor part of a monad T^ν on Set. We shall achieve this by showing that it carries a Kleisli monad structure.

We just defined the action of T^ν on objects, so we only have to define the collection of functors $X \longrightarrow T^\nu X$ and the substitution functions $\mathsf{s}_{X,Y}: \mathsf{C}(X, T^\nu Y) \longrightarrow \mathsf{C}(T^\nu X, T^\nu Y)$. As for the former, we can simply consider the first component of τ^{-1}, i.e. the function $\eta^\nu_X: X \longrightarrow T^\nu X$. Note that $\eta^\nu_X = \phi_X \eta_X$. As for the substitution, we use the metric properties of $T^\nu X$ in order to define a function from $\mathsf{Set}(X, T^\nu Y)$ to $\mathsf{Set}(T^\nu X, T^\nu Y)$. Let f be a function from X to $T^\nu Y$ and note that, because $\alpha_Y: FT^\nu Y \longrightarrow T^\nu Y$ defines an F-algebra structure, we can copair it with f, in order to get an

$X + F$-algebra $[f, \alpha_Y]$ on $T^\nu Y$. This determines a map f' from the initial $X + F$-algebra (TX, i_X) to $(T^\nu Y, [f, \alpha_Y])$, i.e. a function $f'\colon TX \longrightarrow T^\nu Y$. It is not hard to show that such a map is uniformly continuous, and therefore it extends to a map from the metric completion of TX (which is precisely $T^\nu X$) to $T^\nu Y$. We shall write that map as $f^\dagger\colon T^\nu X \longrightarrow T^\nu Y$. Associating to each function f the corresponding $f^\dagger$, we get a map

$$\mathrm{s}_{X,Y}\colon \mathsf{Set}(X, T^\nu Y) \longrightarrow \mathsf{Set}(T^\nu X, T^\nu Y).$$

All we have to do now, is to prove that this assignation satisfies the equations of a Kleisli monad. Let then $f\colon X \longrightarrow T^\nu Y$ and $g\colon Y \longrightarrow T^\nu Z$ be functions.

$\mathrm{s}(f)\eta^\nu_X = f$ This follows easily by the following chain of equalities:

$$\mathrm{s}(f)\eta^\nu_X = f^\dagger\eta^\nu_X = f^\dagger\phi_X\eta_X = f'\eta_X = f;$$

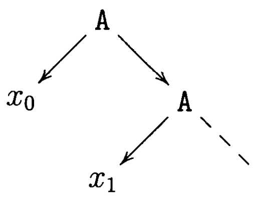
$\mathrm{s}(\eta^\nu_X) = \mathsf{id}_{T^\nu X}$ Using the universal property of ϕ_X, it is enough to show that $\eta^\nu_X{}^\dagger\phi_X = \eta_X$, and this follows trivially by the definitions;

$\mathrm{s}(\mathrm{s}(g)f) = \mathrm{s}(g)\mathrm{s}(f)$ Again by the universal property of ϕ_X, it is enough to prove that precomposition of the two sides of the equation with ϕ_X gives the same result. On the left handside this composite reduces to $(\mathrm{s}(g)f)'$, whereas on the right handside it becomes $\mathrm{s}(g)f'$. To show that they are the same it is enough to notice that they are both algebra morphisms from the initial $X+F$-algebra TX to the same algebra. The computation is an easy diagram chase.

Clearly the argument does not make use of the signature; in fact, that piece of information this is quite irrelevant. All we need to know is that, for each set X, the final $X + F$-coalgebra exists, and that we can apply Barr's construction to show that it is the Cauchy completion of the initial $X + F$-algebra. That yields to the proof of the following result.

Proposition 2.25 *Let F be a polynomial endofunctor and C satisfy the premises of [13, Theorem 3.2]. Then, the map T^{ν} assigning to an object X the carrier of the final $X + F$-coalgebra carries the structure of a monad.*

Not surprisingly, given that we work with final coalgebras, a rank change happens, as we already had in the previous section, and the monad we get is not, in general, finitary. As an example, consider a signature consisting of a binary symbol A. The set $T^{\nu}X$ is the set of finite and infinite binary trees whose nodes are labelled by A and leaves are labelled by elements of X. If this was the colimit of the family of sets $T^{\nu}X_0$, where X_0 is any finite subset of X, then any tree would only involve finitely many different variables; but, for an infinite X and a sequence $x_0, x_1, \ldots$ of different elements in X, we can form the tree



which clearly is in $T^{\nu}X$ and involves infinitely many different variables.

2.4.2 Recursion on Infinite Terms

Although we found this result independently [27], J. Adámek and his research group [1] proved it as well, in a more elegant and general context, only to find out later that Larry Moss had already proved the same things, in yet another way, one year before [49]. This shows that the research in this field is rather active and lively.

Adámek's approach is perhaps the most general and abstract. He and his coauthors considered the work of Elgot and his group [21, 22, 15], who focused on the study of infinite trees and their properties, with particular attention to the solution of recursive equations within that context, and gave a categorical formulation of these concepts, proving that the category T^ν, whenever it exists, is the free iterative monad over the functor F. In order to understand this, we need to give a couple of definitions, taken from [1].

First of all, let's note that an equation in the form of (1.2) can be represented as a map sending each unknown in the set X to the corresponding term in $T(X+Y)$, or, if we allow the right handside to be infinite, in $T^\nu(X+Y)$, where Y is a set of parameters. Such a coalgebra is what we call an *equation morphism.* Note that, again by Lambek's lemma, $T^\nu(X+Y)$ can be split as the sum of X and $Y+FT^\nu(X+Y)$, thus separating variables from parameters and guarded terms. An *F-guarded equation morphism* is an equation morphism $X \longrightarrow T^\nu(X+Y)$ which factors through $Y+FT^\nu X$. This clearly corresponds to an ideal system of equations when $F=F_\Sigma$ for some signature Σ.

A solution to such a system consists of a mapping which associates to each unknown of the system a term depending only on the parameters, in such a way that the result of substituting these terms for the variables in X the two sides of the equations become equal. If we render this diagrammatically, we get that the *solution* to a system $e: X \longrightarrow T^\nu(X+Y)$ is a map $e^\dagger: X \longrightarrow T^\nu Y$ making the following diagram commute:

$$\begin{tikzcd}
X \arrow[rr, "e^\dagger"] \arrow[d, "e"'] & & T^\nu Y \\
FT^\nu(X+Y) \arrow[r, "FT^\nu{[e^\dagger,\eta^\nu_Y]}"'] & FT^{\nu 2}Y \arrow[r, "F\mu^\nu{}_Y"'] & FT^\nu Y. \arrow[u, "\alpha_Y"']
\end{tikzcd} \tag{2.13}$$

Adámek et al. define a functor F *iteratable* if for each X in C the functor $X + F$ has a terminal coalgebra, and, using just the universal property of finality of such coalgebras, they build a substitution map, from which they derive the construction of a monad T^ν (in fact, their substitution theorem implies a slightly stronger version of the Kleisli presentation of a monad). Furthermore, they can prove that for the monad T^ν arising from an iteratable endofunctor F one can solve any F-guarded system of equations. Such monads they call *completely iterative*[1], and T^ν turns out to be the free one over F. This way, they find a counterpart of the definition of a completely iterative theory as given by Elgot.

Our slightly different approach to the subject is motivated by other applications, as we are going to see.

[1] To be precise, one defines a monad (T, η, μ) to be *ideal* when there is a subfunctor $\alpha: T' \longrightarrow T$ such that $[\alpha, \eta]: T' + \mathsf{Id} \longrightarrow T$ is an isomorphism and μ restricts to a natural transformation $\mu': T'T \longrightarrow T'$. In this setting, an equation $X \longrightarrow T(X+Y)$ is guarded when it factors through $T'(X+Y)+Y$, and a monad is completely iterative when all guarded equations have a unique solution.

2.4.3 An Application: the Approximation Lemma

The *generic approximation lemma* [30] is a proof principle for reasoning about functions in a lazy functional programming language (such as Haskell). The approximation lemma itself pertains to lists and states that, given a function

```
approx (n+1) []     = []
approx (n+1) (x:xs) = x:(approx n xs)
```

two lists xs and ys are equal if and only if $\forall n.$approx n xs = approx n ys. Note the lack of a base case: approx 0 x is $\bot$ (i.e., the denotation of undefined) in the denotational model, but, because of non-strictness, approx n x (with $n > 0$) is defined. This principle can be applied to other datatypes such as trees:

```
data Tree a = Leaf a | Node Tree a Tree
approx (n+1) (Leaf x) = Leaf x
approx (n+1) (Node l x r) = Node (approx n l) x (approx n r)
```

Analogously to the previous case, here one shows that two trees t1 and t2 are equal if and only if $\forall n.$ approx n t1 = approx n t2. In [30], the authors prove the generic approximation lemma using the standard denotational semantics of functional programming languages, where types are interpreted as CPO's, programs as continuous functions and recursive datatypes as least fixed points of functors. That is, the correctness of the proof principle depends upon the semantic category chosen; we have already seen the implicit use of $\bot$ in the definition of approx.

We propose an alternative and, we believe, more natural derivation of the approximation lemma which is independent of the particular denotational model chosen. The definition of a polymorphic datatype is usually interpreted as the free monad T over its signature Σ. However, this does not capture laziness, since T consists of only finite terms. Instead, we model such a datatype by the monad T^{ν}. Since $T^{\nu}(X)$ is the final $X + F$-coalgebra and since F is a polynomial endofunctor, $T^{\nu}(X)$ can be calculated as the limit of the following ω^{op}-chain $1 \longleftarrow (X+F)1 \longleftarrow (X+F)^2 1 \longleftarrow \cdots$, as we saw in Proposition 2.9. The universal property of the limit states that two elements x and y of this limit will be equal if and only if, for each n, $\pi_n(x) = \pi_n(y)$ where π_n is the n-th projection. But these projections are precisely the approximation function for the datatype. Notice how the categorical argument replaces the semantic dependency on $\perp$ by making use of a cochain beginning with 1. This establishes the correctness of the generic approximation lemma, independently of any specific denotational model.

Chapter 3

Monads of Terms

In the previous chapters we saw already two different examples of monads building terms over a signature Σ: the monad T of finite terms, and the monad T^ν of finite and infinite terms. Finite terms are known to provide a denotational semantics for the language, by interpreting all term constructors as functions. Infinite terms, instead, allow a study of the behavioural semantics of the language, since they capture any possible evolution of a Σ program. When programming, though, one never really makes use of the full power of this behavioural semantics.

For instance, since the memory locations in a machine are finite, we want our programs to allow only finitely many variables. A monadic semantics for such terms is easily realised by considering the finitary coreflection $\mathsf{Lan}_J(\mathsf{T}^\nu J)$, which is a finitary monad. However, even with this restriction, infinite terms can model very wild behaviours, which cannot even be expressed, if not by giving a detailed description of the full tree. Programs which behave in such an irregular way need being defined in all their evolutions, and this description will typically be infinite, hence it can not be typed into any machine; in fact, it cannot even be fully described.

However, there are ways of modelling infinitary behaviours by means of a finite amount of data, for instance by allowing recursion. This way, we identify a subclass of terms, i.e. the *rational* ones. When implementing lazy functional programming languages, it is useful to model programs by *term graphs*, using term graph rewriting techniques in order to compile the results. Enhancing recursive calls of programs by allowing parameter passing, would give rise to another, more extended, class of terms, i.e. the *algebraic* ones.

All these different syntactic structures share some important features. The main one is definitely that of being closed under substitution. Moreover, one can consider a Σ-algebra structure on each of these sets of terms, and say whether a term is guarded or not. In order to model these features categorically, we shall introduce in this chapter the original notion of an *F-guarded monad* and explore some of its properties.

Syntactically, particular families of terms, like the rational ones, or the algebraic, or even all possibly infinite terms, are usually dealt with by means of their universal properties, expressed either as metric completeness or as closedness under solution of specific kinds of equations (which, again, is ensured by metric arguments). In Section 2.2.2, we saw how the universal property of final coalgebras provides a categorical counterpart to metric completeness. Recall also how, in the end of Section 2.2.1, we got the final coalgebra for a functor by considering a quotient of the coproduct of a generating set of coalgebras. Categorically, this is nothing but a colimit, and, taking the colimit of all coalgebras (which, in fact, we cannot do for size reasons) corresponds to taking the colimit of the generating set. It turns out that, if we restrict ourselves to the appropriate collection of coalgebras, then we can take their colimit and obtain other classes of terms, as the ones mentioned above.

In the second part of this chapter we shall see a theorem on how to build guarded monads as colimits, using the intuition just described.

3.1 F-guarded Monads

In this section, we shall introduce the notion of F-guarded monad. The idea is that F is an endofunctor on a category C, which we may think of as building terms of depth one over a signature. In fact, a lot of concepts make no use of signatures and arities, so we shall give the definitions in more general terms, although our examples will always be drawn from Set and F will always be F_Σ for some Σ.

An F-guarded monad, then, is thought of as a monad which builds some specific type of terms for the signature which F models. If $\mathsf{T} = (T, \eta, \mu)$ is such a monad, then TX will be an algebra of F-terms over X; the unit $\eta\colon X \longrightarrow TX$ will embed variables into our family of terms, and the monad structure (thought of in the Kleisli presentation) will ensure closure under substitution.

Moreover, the F-algebra structure should agree with the monadic one. This can be expressed in many different ways. One is to say that the monad multiplication is an F-algebra homomorphism; another would be to say that

F preserves multiplication. Diagrammatically, we are simply saying that the following square commutes:

$$\begin{tikzcd}
FT^2X \arrow[r, "F\mu_X"] \arrow[d, "\alpha_{TX}"'] & FTX \arrow[d, "\alpha_X"] \\
T^2X \arrow[r, "\mu_X"'] & TX,
\end{tikzcd}$$

where α_X is the algebra structure on TX.

Also, it is reasonable to ask for the algebra structure to respect the action of T on maps, that is to say that α is a natural transformation.

So, this is the intuition underlying our definition. However, we shall introduce the notion formally in a different way. First of all, let's note that to have an F-algebra structure as above is equivalent to have an interpretation of F into T, as made clear by the following result.

Lemma 3.1 *Let $\mathsf{T} = (T, \eta, \mu)$ be a monad and F an endofunctor on a category C. There is a bijection between natural transformations $\tau \colon F \longrightarrow T$ and natural transformations $\alpha \colon FT \longrightarrow T$ making the following diagram commute*

$$\begin{tikzcd}
FTT \arrow[r, "F\mu"] \arrow[d, "\alpha_T"'] & FT \arrow[d, "\alpha"] \\
TT \arrow[r, "\mu"'] & T.
\end{tikzcd} \tag{3.1}$$

Proof. Assume a natural transformation $\tau \colon F \longrightarrow T$ is given. Then, one can define α as the composite $\alpha = \mu\tau_T$. Commutation of (3.1) follows immediately by naturality of τ and the associativity of μ:

$$\begin{tikzcd}
FT^2 \arrow[r, "\tau_{T^2}"] \arrow[d, "F\mu"'] & T^3 \arrow[r, "\mu_T"] \arrow[d, "T\mu"'] & T^2 \arrow[d, "\mu"] \\
FT \arrow[r, "\tau_T"'] & T^2 \arrow[r, "\mu"'] & T.
\end{tikzcd}$$

Conversely, given a natural transformation α, one can define τ as the composite

$$\tau = F \xrightarrow{F\eta} FT \xrightarrow{\alpha} T.$$

To see that the two mappings are actually inverse, consider first $\tau: F \longrightarrow T$; the application of their composite gives a natural transformation $\mu\tau_T F\eta = \mu T(\eta)\tau = \tau$. Conversely, given $\alpha: FT \longrightarrow T$ making (3.1) commute, its image under the reversed composite is the natural transformation $\mu\alpha_T F\eta_T = \alpha F\mu F\eta_T = \alpha F(\mu\eta_T) = \alpha$. □

Definition 3.2 An *F-guarded monad* on C is a 4-tuple $\mathsf{T} = (T, \eta, \mu, \tau)$ such that (T, η, μ) is a monad on C and $\tau: F \longrightarrow T$ is a natural transformation.

A *morphism of F-guarded monads* between (T, η, μ, τ) and (T', η', μ', τ') is a monad morphism ϕ from (T, η, μ) and (T', η', μ') such that $\phi\tau = \tau'$.

Remark 3.3 Notice that the further condition imposed on a monad morphism for it to be an F-guarded one is equivalent, under the bijection described in Lemma 3.1, to commutativity of the following diagram, where α

and α' stay for the composites $\mu\tau_T$ and $\mu'\tau'_{T'}$ respectively

$$\begin{tikzcd}
FT \arrow[r, "F\phi"] \arrow[d, "\alpha"'] & FT' \arrow[d, "\alpha'"] \\
T \arrow[r, "\phi"'] & T'.
\end{tikzcd}$$

In other words, we are requiring that these monad morphisms respect the F-algebra structure which the natural transformation τ induces on the two monads.

Condition (3.1) on α has a very intuitive interpretation, when $F = F_\Sigma$ for some Set-signature. In such cases, $F_\Sigma TX$ is the set of terms of depth one, with variables from TX. The algebra structure $\alpha_X \colon F_\Sigma TX \longrightarrow TX$ "absorbs" the Σ-constructor into the T-terms, and commutativity of (3.1) says that this does not interfere with the multiplication of T. So, if we take a term in $F_\Sigma T^2 X$, we can either absorb the Σ-symbol in the first T-layer and then multiply with μ, or rather multiply under the Σ-context and then absorb the symbol, and get the same result.

Because the action of α makes F vanish, we could think of extending the monad structure of T to FT. Multiplication would just be application of $FT\alpha$ in order to get from $(FT)^2$ to FT^2, and then application of $F\mu$. Unfortunately, we cannot produce the unit η, but this is reasonable, because elements in $F_\Sigma TX$ are guarded by some Σ-symbol, therefore they should not be variables. The way to get around this is to consider the functor $\mathsf{Id} + FT$, where we explicitly add variables.

Lemma 3.4 *Let (T, η, μ, τ) be an F-guarded monad on C. Let $\alpha\colon FT \longrightarrow T$ be the natural transformation induced by τ, such that (3.1) commutes. Let's define*

$$\overline{\eta} = \mathsf{inl}\colon \mathsf{Id} \longrightarrow \mathsf{Id} + FT;$$

and

$$\overline{\mu}\colon \mathsf{Id}{+}FT{+}FT(\mathsf{Id}{+}FT) \xrightarrow{\mathsf{Id}+FT+FT[\eta,\alpha]} \mathsf{Id}{+}FT{+}FT^2 \xrightarrow{\mathsf{Id}+[FT,F\mu]} \mathsf{Id} + FT.$$

Then, the triple $(\mathsf{Id}{+}FT, \overline{\eta}, \overline{\mu})$ is a monad, and the map $[\eta, \alpha]\colon \mathsf{Id}{+}FT \longrightarrow T$ is a monad morphism.

Moreover, given another F-guarded monad (S, η', μ', τ') and a morphism of guarded monads $\psi\colon T \longrightarrow S$, we can derive a monad $(\mathsf{Id} + FS, \overline{\eta'}, \overline{\mu'})$ and ψ induces a monad morphism $\mathsf{Id} + F\psi$ such that $[\eta', \alpha'](\mathsf{Id} + F\psi) = \psi[\eta, \alpha]$.

Proof. Let's write ϕ for $[\eta, \alpha]$. Then, we need to prove the unit and multiplication laws for $(\mathsf{Id} + FT, \overline{\eta}, \overline{\mu})$ to be a monad, and to show that ϕ is a monad morphism.

For the first unit law, it is clear that $\overline{\mu\eta}_{\mathsf{Id}+FT} = \mathsf{Id} + [FT, F(\mu T\phi)]\mathsf{inl}_{\mathsf{Id}+FT} = \mathsf{Id} + FT$. For the second one, we need to show commutativity of

$$\begin{tikzcd}
\mathsf{Id} + FT \arrow[r, "(\mathsf{Id}+FT)\overline{\eta}={[\mathsf{inl},\mathsf{inr}FT\overline{\eta}]}"] \arrow[ddr, bend right, "\mathsf{Id}+FT"'] & \mathsf{Id}{+}FT{+}FT(\mathsf{Id} + FT) \arrow[d, "\mathsf{Id}+FT+FT\phi"] \\
 & \mathsf{Id} + FT{+}FT^2 \arrow[d, "\mathsf{Id}+{[FT,F\mu]}"] \\
 & \mathsf{Id} + FT.
\end{tikzcd}$$

The Id-th component of the two maps is clearly equal; as for the FT-th component, one has that $F(\mu)FT(\phi)FT(\mathsf{inl}) = F(\mu T\eta) = FT$, which proves the two composites equal.

We now turn to the multiplication law, which, exploiting the definitions, becomes the following, diagram:

$$\begin{tikzcd}
(\mathsf{Id}+FT)^2+FT(\mathsf{Id}+FT)^2 \arrow[r, "(\mathsf{Id}+FT)(\mathsf{Id}+FT+FT\phi)"] \arrow[d, "(\mathsf{Id}+FT)^2+FT\phi_{\mathsf{Id}+FT}"'] & (\mathsf{Id}+FT)(\mathsf{Id}+FT+FT^2) \arrow[d, "(\mathsf{Id}+FT)(\mathsf{Id}+{[FT,F\mu]}"'] \\
(\mathsf{Id}+FT)^2+FT^2(\mathsf{Id}+FT) \arrow[d, "(\mathsf{Id}+FT)^2+F\mu_{\mathsf{Id}+FT}"'] & (\mathsf{Id}+FT)^2 \arrow[d, "\mathsf{Id}+FT+FT\phi"'] \\
(\mathsf{Id}+FT)^2+FT(\mathsf{Id}+FT) \arrow[d, "{[(\mathsf{Id}+FT)^2,FT\phi]}"'] & \mathsf{Id}+FT+FT^2 \arrow[d, "\mathsf{Id}+{[FT,F\mu]}"'] \\
\mathsf{Id}+FT+FT^2 \arrow[r, "\mathsf{Id}+{[FT,F\mu]}"'] & \mathsf{Id}+FT.
\end{tikzcd}$$

The $(\mathsf{Id}+FT)^2$-th components of the two maps clearly coincide, so we have to focus on the other two, which we can prove equal by chasing the following diagram

$$\begin{tikzcd}
FT(\mathsf{Id}+FT)^2 \arrow[r, "FT(\mathsf{Id}+FT)\phi"'] \arrow[rr, bend left, "FT(\mathsf{Id}+FT+FT\phi)"] \arrow[d, "FT\phi_{\mathsf{Id}+FT}"'] & FT(\mathsf{Id}+FT)T \arrow[d, "FT\phi_T"'] & FT(\mathsf{Id}+FT+FT^2) \arrow[l, "FT(\phi+FT^2)"] \arrow[d, "FT(\phi(\mathsf{Id}+{[FT,F\mu]}))"'] \\
FT^2(\mathsf{Id}+FT) \arrow[r, "FT^2\phi"'] \arrow[d, "F\mu_{\mathsf{Id}+FT}"'] & FT^3 \arrow[r, "FT\mu"'] \arrow[d, "F\mu_T"'] & FT^2 \arrow[d, "F\mu"'] \\
FT(\mathsf{Id}+FT) \arrow[r, "FT\phi"'] & FT^2 \arrow[r, "F\mu"'] & FT,
\end{tikzcd}$$

where the top right square commutes because, by commutativity of (3.1),

$$\begin{aligned}
\mu\phi_T(\phi + FT^2) &= \mu[\eta_T, \alpha_T]([\eta, \alpha] + FT^2) \\
&= \mu[\eta_T[\eta, \alpha], \alpha_T] \\
&= [[\eta, \alpha], \alpha F\mu] \\
&= [\eta, \alpha](\mathsf{Id} + [FT, F\mu]) \\
&= \phi(\mathsf{Id} + [FT, F\mu]).
\end{aligned}$$

Note how, in the previous diagram, the top half expresses (although under the context of the functor FT) the commutativity of the square expressing the fact that ϕ is a monad morphism.

Consider now the same construction for the F-guarded monad (S, η', μ', τ'), deriving an algebra structure ϕ' which is a monad morphism. It is clear from Remark 3.3 that ψ is such that $\psi\phi = \phi(\mathsf{Id} + F\psi)$, therefore, all we have to prove is that $\mathsf{Id} + F\psi$ is a monad morphism from $\mathsf{Id} + FT$ to $\mathsf{Id} + FS$.

It's trivial to observe that, because $\psi\eta = \eta'$, also $\psi\overline{\eta} = \overline{\eta'}$. The fact that multiplication is preserved is proved by the following diagram

$$\begin{tikzcd}
\mathsf{Id}{+}FH{+}FH(\mathsf{Id}{+}FH) \arrow[r, "(\mathsf{Id}+F\psi)_{\mathsf{Id}+FH}"] \arrow[d, "\mathsf{Id}+FH+FH(\mathsf{Id}+\tau_H)"] & \mathsf{Id}{+}FH{+}FK(\mathsf{Id}{+}FH) \arrow[r, "(\mathsf{Id}+FK)(\mathsf{Id}+F\psi)"] \arrow[d, "\mathsf{Id}+FH+FK(\mathsf{Id}+\tau_H)"] & \mathsf{Id}{+}FK{+}FK(\mathsf{Id}{+}FK) \arrow[d, "\mathsf{Id}+FK+FK(\mathsf{Id}+\tau'_K)"] \\
\mathsf{Id}{+}FH{+}FH(\mathsf{Id}{+}H^2) \arrow[r, "\mathsf{Id}+FH+F\psi_{\mathsf{Id}+H^2}"] \arrow[d, "\mathsf{Id}+FH+FH(\mathsf{Id}+\mu)"] & \mathsf{Id}{+}FH{+}FK(\mathsf{Id}{+}H^2) \arrow[r, "\mathsf{Id}+F\psi+FK(\mathsf{Id}+\psi^2)"] \arrow[d, "\mathsf{Id}+FH+FK(\mathsf{Id}+\mu)"] & \mathsf{Id}{+}FK{+}FK(\mathsf{Id}{+}K^2) \arrow[d, "\mathsf{Id}+FK+FK(\mathsf{Id}+\mu')"] \\
\mathsf{Id}{+}FH{+}FH(\mathsf{Id}{+}H) \arrow[r, "\mathsf{Id}+FH+F\psi_{\mathsf{Id}+H}"] \arrow[dd, "\mathsf{Id}+FH+FH{[\eta,H]}"] & \mathsf{Id}{+}FH{+}FK(\mathsf{Id}{+}H) \arrow[r, "\mathsf{Id}+F\psi+FK(\mathsf{Id}+\psi)"] \arrow[d, "\mathsf{Id}+FH+FK{[\eta,H]}"] & \mathsf{Id}{+}FK{+}FK(\mathsf{Id}{+}K) \arrow[dd, "\mathsf{Id}+FK+FK{[\eta',K]}"] \\
 & \mathsf{Id}{+}FH{+}FKH \arrow[dr, bend left, "\mathsf{Id}+F\psi+FK\psi"] & \\
\mathsf{Id}{+}FH{+}FH^2 \arrow[ur, bend left, "\mathsf{Id}+FH+F\psi_H"] \arrow[rr, "\mathsf{Id}+F\psi+F\psi^2"'] \arrow[d, "\mathsf{Id}+{[FH,F\mu]}"] & & \mathsf{Id}{+}FK{+}FK^2 \arrow[d, "\mathsf{Id}+{[FK,F\mu']}"] \\
\mathsf{Id}{+}FH \arrow[rr, "\mathsf{Id}+F\psi"'] & & \mathsf{Id}{+}FK
\end{tikzcd}$$

which commutes because, by assumption, ψ is a monad morphism and $\psi\tau = \tau'$. This shows that $\mathsf{Id} + F\psi$ is a monad morphism. □

So, for any F-guarded monad T, TX is naturally an $X + F$-algebra. We shall call strong those monads for which this algebra structure is invertible.

Definition 3.5 Let F be an endofunctor on a category C. A *strongly F-guarded monad* on C is an F-guarded monad $\mathsf{T} = (T, \eta, \mu, \tau)$ such that the induced monad morphism

$$[\eta, \mu.\tau_T] \colon \mathsf{Id} + FT \longrightarrow T$$

is an isomorphism.

Strongly guarded monads form a full subcategory of the category of F-guarded monads.

We are now going to show a couple of examples of F-guarded monads, which are meant to show that we really captured the intuition of monads of terms.

If a monad T is F-guarded, then the natural transformation $\tau \colon F \longrightarrow T$ "interprets" F into T. Therefore, it is natural to expect that, by induction, we can interpret all F-terms in T. This is the case, indeed, and τ determines a monad morphism from the free monad T_F to T, which is an F-guarded monad morphism. In fact, the only one.

Proposition 3.6 *Let T be the free monad over an endofunctor F on a category C. Then, T is the initial (strongly) F-guarded monad.*

Proof. Let $\mathsf{T} = (T, \eta, \mu)$ be the free monad over F. Freeness gives a natural transformation $\tau: F \longrightarrow T$. By Lemma 3.1 and Lemma 3.4, τ induces a monad morphism $\phi = [\eta, \mu.\tau_T]: \mathsf{Id} + FT \longrightarrow T$, where $\mathsf{Id} + FT$ is a monad with unit and multiplication $\overline{\eta} = \mathsf{inl}$ and $\overline{\mu}$, respectively.

Next, there is a natural transformation $\mathsf{inr} \circ F\eta: F \longrightarrow \mathsf{Id}+FT$ which, by the freeness of T, corresponds to a monad morphism $\psi: T \longrightarrow \mathsf{Id} + FT$. That $\phi\psi = \mathsf{id}_T$ follows by the freeness of T, once we show that $\phi\psi\tau = \tau$, and this is easily proved by the following chain of equalities:

$$\begin{aligned}\phi\psi\tau &= [\eta, \mu\tau_T]\mathsf{inr}F\eta\\ &= \mu\tau_T F\eta\\ &= \mu T\eta\tau\\ &= \tau.\end{aligned}$$

In the reverse direction, $\psi\phi\mathsf{inl} = \overline{\eta}$, since $\psi\phi$ is a monad morphism, and

$$\begin{aligned}\psi\phi\mathsf{inr} &= \psi\mu\tau_T\\ &= \overline{\mu}(\mathsf{Id} + FT)(\psi)\psi_T\tau_T\\ &= \overline{\mu}(\mathsf{Id} + FT)(\psi)\mathsf{inr}F\eta_T\\ &= \overline{\mu}\mathsf{inr}F(T(\psi)\eta_T)\\ &= \mathsf{inr}F(\mu T\phi T(\psi)\eta_T)\\ &= \mathsf{inr}F(\mu\eta_T)\\ &= \mathsf{inr},\end{aligned}$$

where the second equality follows from the fact that ψ is a monad morphism, the fifth from the definition of $\overline{\mu}$, and the sixth by the fact that $\phi\psi = \mathsf{id}_T$, as just shown.

Hence, $\psi\phi$ is the identity on $\mathsf{Id} + FT$ and T is a strongly F-guarded monad. Given any other (strongly) F-guarded monad $\mathsf{T}' = (T', \eta', \mu', \tau')$, freeness of T over F and the transformation $\tau' \colon F \longrightarrow T'$ give a unique monad morphism $! \colon \mathsf{T} \longrightarrow \mathsf{T}'$ such that $!\tau = \tau'$, so ! is also a guarded monad morphism. Since a guarded monad morphism is a monad morphism, uniqueness of ! is ensured by the freeness of T, and the result is proved. □

An easier proof is possible when T pointwise is computed as the carrier of the initial $X + F$-algebra. In such a setting, T is the initial algebra for the endofunctor $(\mathsf{Id} + F \circ -) \colon [\mathsf{C}, \mathsf{C}] \longrightarrow [\mathsf{C}, \mathsf{C}]$, and, since all initial algebras are isomorphisms, we get T is isomorphic to $\mathsf{Id} + FT$. Initiality follows, since, by Lemma 3.1 every F-guarded monad is an $(\mathsf{Id} + F \circ -)$-algebra.

If an F-guarded monad T is to be seen as a monad of terms, or, more generally, of syntactic structures over F, then we can expect to be able to map all terms in TX to some (possibly infinite) terms in $T^{\nu}X$, which, as we saw, is the set of all possible terms, thought of as the evolutions of an F-system. The map should be obtained by mapping each term in TX, i.e. each program, to its behaviour, which in turn can be derived by considering the unfolding of the system and corecursively building a bisimilar term in $T^{\nu}X$. In order to do that, though, we need to have a coalgebra structure on TX, i.e. we need T to be strongly F-guarded.

Proposition 3.7 *Let F be an iteratable endofunctor on a category C, and let $\mathsf{T}^{\nu} = (T^{\nu}, \eta^{\nu}, \mu^{\nu})$ be the free completely iterative monad over F. Then, T^{ν} is final amongst all strongly F-guarded monads.*

Proof. Recall from Section 2.4 that an iteratable endofunctor F is such that $X+F$ has a final coalgebra for each X in C, and, in that case, the functor T^{ν} mapping X to the carrier of the final $X+F$-coalgebra has a monad structure. Furthermore, T^{ν} is the free completely iterative monad over F, as shown in [1].

It is clear that the pointwise coalgebra structure on each $T^{\nu}X$ determines on T^{ν} a coalgebra structure $c\colon T^{\nu} \longrightarrow \mathsf{Id} + FT^{\nu}$ for the endofunctor $(\mathsf{Id} + F \circ -)\colon \mathsf{End}(\mathsf{C}) \longrightarrow \mathsf{End}(\mathsf{C})$, and this is an isomorphism, with inverse $[\eta^{\nu}, \alpha^{\nu}]$, as shown by pointwise applying Lambek's lemma.

From the substitution theorem [1, Theorem 2.17], it also follows that the multiplication μ^{ν} of T^{ν} satisfies (3.1), therefore T^{ν} is strongly F-guarded, where $\tau^{\nu}\colon F \longrightarrow T^{\nu}$ is derived via Lemma 3.1 from the F-algebra structure α^{ν}. In order to prove that it is the final one, let's just note that, given any strongly F-guarded monad $\mathsf{S} = (S, \eta, \mu, \tau)$ and deriving the function $\alpha = \mu\tau_S$ via Lemma 3.1, the inverse c^S of $[\eta, \alpha]$ determines on S an $(\mathsf{Id} + F \circ -)$-coalgebra structure. Since T^{ν} is final among such coalgebras, there is a unique $(\mathsf{Id} + F \circ -)$-coalgebra homomorphism $\phi\colon S \longrightarrow T^{\nu}$.

We want to show that ϕ is an F-guarded monad morphism, i.e. that it is a monad morphism and that $\phi\tau^{\nu} = \tau$. If we achieve this, we have the result, as any morphism between strongly F-guarded monads is necessarily a homomorphism of $(\mathsf{Id} + F \circ -)$-coalgebras, therefore uniqueness follows by finality of T^{ν} among them. Note how here it is essential that the monads are *strongly* guarded, in order to reverse the naturally induced algebra structure and obtain a coalgebra one.

So, all we have to show is that

1. $\phi\eta = \eta^{\nu}$;

2. $\phi\mu = \mu^{\nu}T(\phi)\phi_S$;

3. $\phi\tau = \tau^{\nu}$.

First, note that, being both coalgebra structures invertible, we get that ϕ is also an $(\mathsf{Id} + F \circ -)$-algebra homomorphism, hence, for each X in C, we have

$$\begin{tikzcd}
X + FX \arrow[r, "\mathsf{id}+F\phi_X"] \arrow[d, "{[\eta_X, \alpha_X]}"'] & X + FT^{\nu}X \arrow[d, "{[\eta^{\nu}_X, \alpha^{\nu}_X]}"] \\
SX \arrow[r, "\phi_X"'] & T^{\nu}X.
\end{tikzcd}$$

From this, precomposing with the left and the right injection into $X + FSX$, we get at once that $\phi_X\eta_X = \eta^{\nu}{}_X$, and also that $\phi_X\tau_X = \phi_X\alpha_X F\eta_X = \alpha^{\nu}_X F(\phi\eta_X) = \alpha^{\nu}_X F\eta^{\nu}_X = \tau^{\nu}_X$. So, all we have to show is that equality 2 above holds.

This follows by chasing the diagrams below, which prove that both composites are $(\mathsf{Id}+F \circ -)$-coalgebra homomorphisms from the same $(\mathsf{Id}+F \circ -)$-

coalgebra to the final such, $c\colon T^\nu \longrightarrow \mathsf{Id} + FT^\nu$, hence having to be equal.

$$\begin{tikzcd}
S^2 \arrow[r, "\phi_S"] \arrow[d, "c^S_S"'] & T^\nu S \arrow[r, "T^\nu\phi"] \arrow[d, "c_S"'] & T^{\nu 2} \arrow[r, "\mu^\nu"] \arrow[d, shift right, "c_{T^\nu}"'] & T^\nu \arrow[ddd, shift left, "c"] \\
S{+}FS^2 \arrow[r, "S+F\phi_S"] \arrow[d, "c^S+FS^2"'] & S{+}FT^\nu S \arrow[r, "\phi+FT^\nu\phi"] \arrow[d, "c^S+FT^\nu S"'] & T^\nu{+}FT^{\nu 2} \arrow[u, shift right, "{[\eta^\nu_{T^\nu},\alpha^\nu_{T^\nu}]}"'] \arrow[d, "c+FT^{\nu 2}"'] \arrow[ddr, bend left=10, "{[c,\mathsf{inr}F\mu^\nu]}"] & \\
\mathsf{Id}{+}FS{+}FS^2 \arrow[r, "\mathsf{Id}+FS+F\phi_S"] \arrow[d, "\mathsf{Id}+{[F\eta_S,FS^2]}"'] & \mathsf{Id}{+}FS{+}FT^\nu S \arrow[r, "\mathsf{Id}+F\phi+FT^\nu\phi"] \arrow[d, "\mathsf{Id}+{[F\eta^\nu_S,FTS]}"'] & \mathsf{Id}{+}FT^\nu{+}FT^{\nu 2} \arrow[d, "\mathsf{Id}+{[F\eta^\nu_{T^\nu},FT^{\nu 2}]}"'] \arrow[dr, bend left=10, "\mathsf{Id}+{[FT^\nu,F\mu^\nu]}"] & \\
\mathsf{Id}{+}FS^2 \arrow[r, "\mathsf{Id}+F\phi_S"'] & \mathsf{Id}{+}FT^\nu S \arrow[r, "\mathsf{Id}+FT^\nu\phi"'] & \mathsf{Id}{+}FT^{\nu 2} \arrow[r, "\mathsf{Id}+F\mu^\nu"'] & \mathsf{Id}{+}FT^\nu \arrow[uuu, shift left, "{[\eta^\nu,\alpha^\nu]}"]
\end{tikzcd}$$

$$\begin{tikzcd}
S^2 \arrow[r, "\mu"] \arrow[d, shift right, "c^S_S"'] & S \arrow[r, "\phi"] \arrow[ddd, shift left, "c^S"] & T^\nu \arrow[ddd, "c"] \\
S + FS^2 \arrow[u, shift right, "{[\eta_S,\alpha_S]}"'] \arrow[d, "c^S+FS^2"'] \arrow[ddr, bend left=10, "{[c^S,\mathsf{inr}F\mu]}"] & & \\
\mathsf{Id} + FS + FS^2 \arrow[d, "\mathsf{Id}+{[F\eta_S,FS^2]}"'] \arrow[dr, "\mathsf{Id}+{[FS,F\mu]}"'] & & \\
\mathsf{Id} + FS^2 \arrow[r, "\mathsf{Id}+F\mu"'] & \mathsf{Id} + FS \arrow[r, "\mathsf{Id}+F\phi"'] \arrow[uuu, shift left, "{[\eta,\alpha]}"] & \mathsf{Id} + FT^\nu
\end{tikzcd}$$

□

3.2 Monads as Pointwise Colimits

In this section, we provide an original result: a theorem for producing monads as pointwise colimits. The reason for seeking such a result is, as explained at the beginning of this chapter, that in the practice we often deal with families of infinite terms which can be obtained by "putting together" appropriate coalgebras, i.e. taking the colimit of a suitable subcategory of the category of coalgebras for an endofunctor. This is somehow a generalisation of the construction of the final coalgebra shown at the end of Section 2.2.1.

The spirit of the result, then, is to consider an endofunctor F on a category C and, for each object X in C, a subcategory $\mathcal{I}X$ of $X + F$-coalgebras. The aim is to show that the mapping $X \longmapsto \operatorname{colim} \mathcal{I}X$ forms the functor part of an F-guarded monad. In order to get the result, though, we need a few assumptions, and we shall have to restrict ourselves to locally presentable categories, but we shall discuss the assumptions in due time.

The result was first presented in [26], and the motivation for it was that we could provide a unified method for constructing a monad for rational terms and a monad for term graphs, as well as deriving some of their basic properties. These examples, which we shall explore in detail in the next chapter, show that the rather clumsy assumptions we will have to make are indeed reasonable.

Although this is the idea, our $\mathcal{I}X$ will not be a category of coalgebras; rather, a category with a (forgetful) functor $U_X : \mathcal{I}X \longrightarrow \mathsf{C}$, and we shall get a monad on C by considering for each X the object $\operatorname{colim} U_X$. Formally, we shall need to work with a weak 2-categorical version of slice categories, which are defined as follows.

Definition 3.8 Let C be a 2-category (see Example 1.42-2) and X an object of C. The *lax slice 2-category* Lax_X has as objects maps $f\colon Y \longrightarrow X$. Arrows are given by

$$\mathsf{Lax}_X(f,g) = \{\langle h,\alpha\rangle | \alpha\colon f \Rightarrow gh\}$$

If $\langle h,\alpha\rangle\colon f \longrightarrow g$ and $\langle k,\beta\rangle\colon g \longrightarrow l$ are morphisms in Lax_X, we write their composite as $\langle kh, \beta\Diamond\alpha\rangle\colon f \xrightarrow{\alpha} gh \xrightarrow{\beta h} lkh$.

Given $\langle h,\alpha\rangle\colon f \longrightarrow g$ and $\langle h',\alpha'\rangle\colon f \longrightarrow g$, a 2-cell $\langle h,\alpha\rangle \longrightarrow \langle h',\alpha'\rangle$ in Lax_X consists of a 2-cell $\theta\colon h \longrightarrow h'$ in C such that $\alpha' = g\theta.\alpha$, where $\alpha.\beta$ stands for the vertical composition of 2-cells α and β.

Notice the usual definition of a slice category C/X is the lax slice category on the 2-category obtained by adding only the identity 2-cells to C. Our first use of slice categories is to state the following result.

Lemma 3.9 *Consider* Cat *as a base 2-category and build the category* Lax_C *over a cocomplete category* C. *Then, the assignment to each* $F\colon \mathsf{A} \longrightarrow \mathsf{C}$ *of its colimit* $\mathrm{colim}\,F$ *defines a 2-functor* $\mathrm{colim}\colon \mathsf{Lax}_\mathsf{C} \longrightarrow \mathsf{C}$, *where* C *is made into a 2-category by allowing only identity 2-cells.*

Proof. We already know the action of the functor colim on objects: given a functor $F\colon \mathsf{A} \longrightarrow \mathsf{C}$ we map it to the object $\mathrm{colim}\,F$ in C, which exists because C is cocomplete. In particular, for X is A we will denote the colimiting map by $\overline{X}\colon FX \longrightarrow \mathrm{colim}\,F$. Given two such objects and a map between them in $\mathsf{Lax}_{\mathsf{C}}$ as shown below,

$$\begin{tikzcd}
\mathsf{A} \arrow[rr, "H"] \arrow[dr, "F"'] & {} & \mathsf{B} \arrow[dl, "G"] \\
& \mathsf{C} &
\arrow[from=1-2, to=2-2, phantom, "\Rightarrow^{\alpha}"]
\end{tikzcd}$$

the composites $\overline{HX}\alpha_X$ (X in A) of the X-th component of α with the HX-th component of the colimiting cocone over the diagram G, determine a cocone over F, since, for $f\colon X \longrightarrow Y$ in A, one has $\overline{HY}\alpha_Y Ff = \overline{HY}GHf\alpha_X = \overline{HX}\alpha_X$. Such a cocone then determines a unique map from $\operatorname{colim} F$ to $\operatorname{colim} G$, which we take to be the image of $\langle H, \alpha\rangle$ along colim.

To see that the assignment preserves composition, consider $\langle H, \alpha\rangle\colon F \longrightarrow G$ and $\langle K, \beta\rangle\colon G \longrightarrow L$. Then, precomposing with $\overline{X}\colon FX \longrightarrow \operatorname{colim} F$, one has that

$$\begin{aligned}
\operatorname{colim}(\beta\Diamond\alpha)\overline{X} &= \overline{KHX}\beta_{HX}\alpha_X \\
&= \operatorname{colim}\beta\,\overline{HX}\alpha_X \\
&= \operatorname{colim}\beta\operatorname{colim}\alpha\overline{X}.
\end{aligned}$$

Hence, because the family $\overline{X}$ is jointly epic, $\operatorname{colim}(\beta\Diamond\alpha) = \operatorname{colim}\beta\operatorname{colim}\alpha$.

Finally, in order for colim to be a 2-functor, we need it to map 2-cells to 2-cells, but in C the only 2-cells we have are the identity ones, so this requirement amounts to say that, given a 2-cell $\beta\colon \langle H, \alpha\rangle \longrightarrow \langle H', \alpha'\rangle\colon F \longrightarrow G$ (i.e. $G\beta.\alpha = \alpha'$), we have $\operatorname{colim}\alpha = \operatorname{colim}\alpha'$. But this is again easily proved by precomposing with the family $\overline{X}\colon FX \longrightarrow \operatorname{colim} F$. In fact, we have

$$\begin{aligned}\operatorname{colim} \alpha\overline{X} &= \overline{HX}\alpha_X \\ &= \overline{H'X}G(\beta_X).\alpha_X \\ &= \overline{H'X}\alpha'_X \\ &= \operatorname{colim} \alpha'\overline{X},\end{aligned}$$

where the second equality holds because $\overline{H'X}$ and $\overline{HX}$ are maps in a cocone over G. □

3.2.1 Kleisli Monoids

In the applications of our theorem, we shall consider, for each object X in C, a subcategory $\mathcal{I}X$ of the category of F-coalgebras, together with a restriction of the forgetful functor U_X. We shall then get the action of the monad T we want to define by mapping each X to $\operatorname{colim} U_X$. This is the same as defining T as the composite

$$\begin{tikzcd} \mathsf{C} \arrow[r, "\langle \mathcal{I}, U \rangle"] & \mathsf{Lax}_{\mathsf{C}} \arrow[r, "\operatorname{colim}"] & \mathsf{C} \end{tikzcd} \tag{3.2}$$

where $\langle \mathcal{I}, U \rangle X = U_X\colon \mathcal{I}X \longrightarrow \mathsf{C}$. We shall prove that it is a Kleisli triple, thus getting our result.

Well, in fact this is not exact. For our construction to work, we shall have to restrict our attention to finitely presentable categories.

Assumption 1: C is a locally λ-presentable category.

We shall construct λ-accessible monads under the equivalence between $\mathsf{Mon}(\mathsf{C})_\lambda$ and monoids in $\mathcal{M}on([\mathsf{C}_\lambda, \mathsf{C}])$ described in Remark 1.32, by deriving a monoid structure in $[\mathsf{C}_\lambda, \mathsf{C}]$ for the restriction of T to C_λ (which we shall still write as T, with an abuse of notation), and this we shall do by defining for it the structure of a *Kleisli monoid.* The generalisation to any regular cardinal λ is straightforward.

Definition 3.10 Let C be an lλp category. A *Kleisli monoid* on the category $[\mathsf{C}_\lambda, \mathsf{C}]$ is a triple consisting of

- a function T assigning to each object X in C_λ an object TX in C;
- for each X in C_λ a map $\eta_X : X \longrightarrow TX$ in C;
- for any pair X and Y of λ-presentable objects, a *substitution* function $\mathrm{s}_{X,Y} : \mathsf{C}(X, TY) \longrightarrow \mathsf{C}(TX, TY)$;

satisfying the following conditions:

1. $\mathrm{s}_{X,X}(\eta_X) = \mathsf{id}_{TX}$;
2. $\mathrm{s}_{X,Y}(f)\eta_X = f$;
3. $\mathrm{s}_{X,Z}(\mathrm{s}_{Y,Z}(g)f) = \mathrm{s}_{Y,Z}(g)\mathrm{s}_{X,Y}(f)$.

As for the case of Kleisli triples, we shall omit subscripts to the substitution functions whenever possible. The definition does not substantially differ from that of a Kleisli triple for a category C, and, not surprisingly, it gives an equivalent characterisation of a monoid in the category $[\mathsf{C}_\lambda, \mathsf{C}]$, as shown by the following result.

Proposition 3.11 *If C is an $l\lambda p$ category, there's a bijective correspondence between Kleisli monoids on the category $[\mathsf{C}_\lambda, \mathsf{C}]$ and monoids on the same category.*

Proof. Given a monoid in $[\mathsf{C}_\lambda, \mathsf{C}]$, this induces a λ-accessible monad on C by Remark 1.32, and, by representing it as a Kleisli triple and considering its restriction to C_λ, we get precisely the desired structure.

Conversely, given T, η and s as in Definition 3.10, we can extend T to a functor from C_λ to C by defining $T(f) = \mathrm{s}(\eta_Y f)$ for any $f{:}\, X \longrightarrow Y$ in C_λ. That f preserves identities follows from equation 1. Furthermore, given $f{:}\, X \longrightarrow Y$ and $g{:}\, Y \longrightarrow Z$ in C_λ, we have $T(g)T(f) = \mathrm{s}(\eta_Z g)\mathrm{s}(\eta_Y f) = \mathrm{s}(\mathrm{s}(\eta_Z g)\eta_Y f) = \mathrm{s}(\eta_Z gf) = T(gf)$, hence T is a functor.

In order to show a unit for T, we need to give a natural transformation from the inclusion functor $J{:}\, \mathsf{C}_\lambda \longrightarrow \mathsf{C}$ to T. That is pointwise defined by $\eta_X{:}\, X \longrightarrow TX$. Naturality follows from equation 2, when replacing f by $\eta_Y f$.

We also have to define a multiplication $\mu{:}\, T \otimes T \longrightarrow T$, where $T \otimes T$ in $[\mathsf{C}_\lambda, \mathsf{C}]$ is given by the functor $(\mathsf{Lan}_{J_\lambda} T)T$, as we saw in Example 1.18-3. In order to define μ_X, let's express TX as a colimit of λ-presentable objects: $TX = \operatorname{colim} X_d$ (d in D), with colimiting maps $\bar{d}{:}\, X_d \longrightarrow TX$. Then, $(\mathsf{Lan}_{J_\lambda} T)TX$ explicitly computes as $\operatorname{colim} TX_d$, with colimiting maps $\tilde{d}{:}\, TX_d \longrightarrow \mathsf{Lan}_{J_\lambda} T(TX)$. For any X_d, the action of s on $\bar{d}{:}\, X_d \longrightarrow TX$, gives a map $\mathrm{s}(\bar{d}){:}\, TX_d \longrightarrow TX$. Given $k{:}\, d \longrightarrow d'$ in D, one has

$$\mathrm{s}(\overline{d'})T\phi = \mathrm{s}(\overline{d'})\mathrm{s}(\eta_{X_{d'}}\phi) = \mathrm{s}(\mathrm{s}(\overline{d'})\eta_{X_{d'}}\phi) = \mathrm{s}(\overline{d'}\phi) = \mathrm{s}(\bar{d}),$$

therefore the maps $\mathrm{s}(\bar{d})$ define a cocone over TX_d (d in D), and induce a map

$\mu_X\colon (T\otimes T)X = \operatorname{colim} TX_d \longrightarrow TX$ such that, for each d,

$$\mu_X\tilde{d} = \mathrm{s}(\overline{d}). \tag{3.3}$$

Naturality for μ means that, for any $f\colon X \longrightarrow Y$ in C_λ, $\mu_Y(T\otimes T)f = T(f)\mu_X$. We know that $(T\otimes T)X$ is determined by the colimiting cocone $\tilde{d}\colon TX_d \longrightarrow (\mathsf{Lan}_J T)TX$, hence we have the equality if we show that, for any d in D, $\mu_Y(T\otimes T)f\tilde{d} = T(f)\mu_X\tilde{d}$. The map $(T\otimes T)f$ is determined by considering, for each composite $X_d \xrightarrow{\overline{d}} TX \xrightarrow{T(f)} TY$, where $\overline{d}$ is a colimiting map for TX, the factorisation through the λ-filtered colimit $\overline{c}\colon Y_c \longrightarrow TY$ with Y_c λ-presentable:

$$\begin{tikzcd}
X_d \arrow[r, "\overline{d}"] \arrow[d, "f_{dc}"'] & TX \arrow[d, "Tf=\mathrm{s}(\eta_Y f)"] \\
Y_c \arrow[r, "\overline{c}"'] & TY.
\end{tikzcd}$$

$(T\otimes T)f$ is then determined by the cocone $\tilde{c}T(f_{dc})\colon TX_d \longrightarrow (T\otimes T)Y$. It is now easy to show, using equations 1–3 and (3.3), that

$$\begin{aligned}
\mu_Y(T\otimes T)f\tilde{d} &= \mu_Y\tilde{c}T(f_{dc}) \\
&= \mathrm{s}(\overline{c})\mathrm{s}(\eta_{Y_c}f_{dc}) \\
&= \mathrm{s}(\mathrm{s}(\overline{c})\eta_{Y_c}f_{dc}) \\
&= \mathrm{s}(\overline{c}f_{dc}) \\
&= \mathrm{s}(\mathrm{s}(\eta_Y f)\overline{d}) \\
&= \mathrm{s}(\eta_Y f)\mathrm{s}(\overline{d}) \\
&= T(f)\mu_X\tilde{d}.
\end{aligned}$$

We now have to show that the equations of a monoid are satisfied; that is, $\mu(T \otimes \eta) = \rho$, $\mu(\eta \otimes T) = \lambda$ and $\mu(T \otimes \mu) = \mu(\mu \otimes T)\alpha$, where α, λ and ρ are the natural isomorphisms of the monadic structure (see Definition 1.17).

The explicit formulation of the first equation is $\mu(\mathsf{Lan}_J T)\eta = \mathsf{id}_T$. It is easy to see that, since $\eta_X \colon X \longrightarrow TX$ is a map in the cocone of all λ-presentable objects over TX, the map $(\mathsf{Lan}_J T)\eta_X \colon (\mathsf{Lan}_J T)X = TX \longrightarrow (\mathsf{Lan}_J T)TX$ is in fact the corresponding $\widetilde{\eta_X}$ in the cocone defining $(\mathsf{Lan}_J T)TX$. Therefore, one has $\mu_X(\mathsf{Lan}_J T)\eta_X = \mu_X\widetilde{\eta_X} = \mathsf{s}(\eta_X) = \mathsf{id}_{TX}$.

The second equation becomes $\mu(\mathsf{Lan}_J \eta)_T = \mathsf{id}_T$, and, in order to show this equality on an object X, we precompose with an arbitrary colimiting map $\overline{d} \colon X_d \longrightarrow TX$. Notice that $(\mathsf{Lan}_J \eta)_{TX}$ is computed as the mediating map induced by the cocone

$$\begin{tikzcd} X_d \arrow[r, "\eta_{X_d}"] & TX_d \arrow[r, "\widetilde{d}"] & (\mathsf{Lan}_J T)TX. \end{tikzcd}$$

Hence, we have, for each d, the equation $(\mathsf{Lan}_J \eta)_{TX}\overline{d} = \widetilde{d}\eta_{X_d}$, from which we derive the following chain of equalities, proving the equation to hold:

$$\begin{aligned} \mu_X(\mathsf{Lan}_J \eta)_{TX}\overline{d} &= \mu_X\widetilde{d}\eta_{X_d} \\ &= \mathsf{s}(\overline{d})\eta_{X_d} \\ &= \overline{d}. \end{aligned}$$

The third equation, is slightly more complicated to get. First of all, we should note that the two objects $\mathsf{Lan}_J((\mathsf{Lan}_J T)T)TX$ and $(\mathsf{Lan}_J T)(\mathsf{Lan}_J T)TX$ are naturally isomorphic. The first is derived as a colimit, with colimiting maps $\widehat{d} \colon (\mathsf{Lan}_J T)TX_d \longrightarrow \mathsf{Lan}_J((\mathsf{Lan}_J T)T)TX$, where $\overline{d}$ is a colimiting map $X_d \longrightarrow TX$. The second is a colimit $\widetilde{c} \colon TZ_c \longrightarrow (\mathsf{Lan}_J T)(\mathsf{Lan}_J T)TX$, where the maps $\overline{c}\colon Z_c \longrightarrow (\mathsf{Lan}_J T)TX$ determine $(\mathsf{Lan}_J T)TX$ as a colimit of objects in C_λ. Notice, though, that, because Z_c is a λ-presentable object, the map $\overline{c}$ factorises through the λ-filtered colimit defining $(\mathsf{Lan}_J T)TX$, i.e. through some $\widetilde{d}\colon X_d \longrightarrow (\mathsf{Lan}_J T)TX$, say as $\overline{c} = \widetilde{d}c'$. Conversely, each map of a λ-presentable object into TX_d determines a map in the cocone over $(\mathsf{Lan}_J T)TX$ by postcomposition with $\widetilde{d}$. This correspondence between the two cocones determines an isomorphism for which the following diagrams commute:

$$\begin{tikzcd}
Z_c \arrow[r, "c'"] \arrow[dr, "\overline{c}"'] & TX_d \arrow[d, "\widetilde{d}"] \\
 & (\mathsf{Lan}_J T)TX
\end{tikzcd}
\qquad
\begin{tikzcd}
(\mathsf{Lan}_J T)TX_d \arrow[d, "\widehat{d}"'] & TZ_c \arrow[l, "\widetilde{c'}"'] \arrow[d, "\widetilde{c}"] \\
\mathsf{Lan}_J((\mathsf{Lan}_J T)T)TX & (\mathsf{Lan}_J T)(\mathsf{Lan}_J T)TX \arrow[l, dash, "\cong"]
\end{tikzcd}$$

With these notations, the two maps $\mu_X(\mathsf{Lan}_J T)\mu_X$ and $\mu_X(\mathsf{Lan}_J \mu)TX$ are defined by the following diagrams,

$$\begin{tikzcd}
TZ_c \arrow[r, "\widetilde{c'}"'] \arrow[rr, bend left, "\mathrm{s}(c')"] & (\mathsf{Lan}_J T)TX_d \arrow[r, "\mu_{X_d}"'] \arrow[d, "\widehat{d}"'] & TX_d \arrow[d, "\widetilde{d}"'] \arrow[dr, "\mathrm{s}(\overline{d})"] & \\
 & \mathsf{Lan}_J((\mathsf{Lan}_J T)T)TX \arrow[r, "(\mathsf{Lan}_J \mu)TX"'] & (\mathsf{Lan}_J T)TX \arrow[r, "\mu_X"'] & TX
\end{tikzcd}$$

$$\begin{tikzcd}
TZ_c \arrow[r, "T\mu_{cd}=\mathrm{s}(\eta_{X_d}\mu_{cd})"] \arrow[d, "\widetilde{c}"'] & TX_d \arrow[d, "\widetilde{d}"'] \arrow[dr, "\mathrm{s}(\overline{d})"] & \\
(\mathsf{Lan}_T J)(\mathsf{Lan}_T J)TX \arrow[r, "(\mathsf{Lan}_J T)\mu_X"'] & (\mathsf{Lan}_J T)TX \arrow[r, "\mu_X"'] & TX
\end{tikzcd}$$

where the map μ_{cd} is derived as in the analogous case above, and is such that $\overline{d}\mu_{cd} = \mu_X\overline{c}$. By precomposing them with $\widetilde{c}\colon TZ_c \longrightarrow (\mathsf{Lan}_J T)(\mathsf{Lan}_J T)TX$, we get two equal maps, as shown by the following chain of equations:

$$\begin{aligned}
\mu_X(\mathsf{Lan}_J T)\mu_X\widetilde{c} &= \mathrm{s}(\bar{d})\mathrm{s}(\eta_{X_d}\mu_{cd}) \\
&= \mathrm{s}(\mathrm{s}(\bar{d})\eta_{X_d}\mu_{cd}) \\
&= \mathrm{s}(\bar{d}\mu_{cd}) = \mathrm{s}(\mu_X\bar{c}) \\
&= \mathrm{s}(\mu_X\widetilde{\bar{d}}c') = \mathrm{s}(\mathrm{s}(\bar{d})c') \\
&= \mathrm{s}(\bar{d})\mathrm{s}(c') \\
&= \mu_X(\mathsf{Lan}_J\mu)TX.
\end{aligned}$$

So, we have a monoid in $[\mathsf{C}_\lambda, \mathsf{C}]$. We now need to show that the two constructions are mutually inverse, but this is obvious, because taking the left Kan extension along J and then restricting to λ-presentable objects does not affect T, and the proof goes on as in the second part of the proof of Theorem 1.28. □

So, we are now ready to approach the construction of the monad T. Let's consider on C a 2-categorical structure where the only 2-cells are the identities.

Assumption 2: Let $\langle \mathcal{I}, U\rangle : \mathsf{C} \longrightarrow \mathsf{Lax}_{\mathsf{C}}$ be a 2-functor.

This means that, for any X in C, we have a category $\mathcal{I}X$ and a functor $U_X : \mathcal{I}X \longrightarrow \mathsf{C}$, and for any map $f : X \longrightarrow Y$ in C a functor $\mathcal{I}f : \mathcal{I}X \longrightarrow \mathcal{I}Y$ and a natural transformation $j^f : U_X \longrightarrow U_Y\mathcal{I}f$, such that

$$\begin{tikzcd}
\mathcal{I}X \arrow[rr, "\mathcal{I}f"] \arrow[dr, "U_X"'] & \arrow[d, phantom, "\Rightarrow j^f"] & \mathcal{I}Y \arrow[dl, "U_Y"] \\
& \mathsf{C}. &
\end{tikzcd} \tag{3.4}$$

We define the functor $T: \mathsf{C}_\lambda \longrightarrow \mathsf{C}$ by considering the restriction of (3.2) to finitely presentable objects. We shall give for T a Kleisli monoid structure, hence deducing, by Proposition 3.11, a monoid in $[\mathsf{C}_\lambda, \mathsf{C}]$, and, by Remark 1.32, that its left Kan extension along the inclusion functor $J: \mathsf{C}_\lambda \longrightarrow \mathsf{C}$ gives a λ-accessible monad on C.

3.2.2 Defining the Functions

We already explained the action of T on each finitely presentable object of C: we put $TX = \operatorname{colim} U_X$.

We are now going to define, for λ-presentable objects X and Y, maps η_X and substitution functions $\mathrm{s}_{X,Y}$ as in Definition 3.10. In the next section, we shall prove them to satisfy equations 1–3, hence defining a Kleisli monoid on $[\mathsf{C}_\lambda, \mathsf{C}]$.

In doing this, we shall have to introduce some further assumptions on the $\langle \mathcal{I}, U \rangle$. For example, in order to define the unit map, it is enough to have

Assumption 3: For each X in C_λ, there is an object i_X in $\mathcal{I}X$ such that $U_X(i_X) = X$.

Then, we can define η_X to be the following colimiting map:

$$\overline{i_X}: X = U_X(i_X) \longrightarrow \operatorname{colim} U_X = TX.$$

Let's now consider a map $f: X \longrightarrow TY$. The leading intuition behind the substitution map $\mathrm{s}(f): TX \longrightarrow TY$ is that a term t in TX is represented by a coalgebra where some states are mapped to variables in X. When performing the substitution, these states are replaced by the carrier of the coalgebras representing the terms specified by f. This way, we get a new $Y+F$-coalgebra representing the term $\mathrm{s}(f)(t)$. In order to make this precise, we introduce a new notion: that of a lifting.

Definition 3.12 A *lifting* $\langle L, ()^L\rangle$ for a 2-functor $\langle \mathcal{I}, U\rangle$ assigns to each object X in C a lax natural transformation

$$\langle L, ()^L\rangle : \langle \mathcal{I}, U\rangle \circ U_X \Longrightarrow \mathsf{K}_{\langle \mathcal{I},U\rangle X} : \mathcal{I}X \longrightarrow \mathsf{Lax}_{\mathsf{C}}.$$

Remark 3.13 The definition of lifting, as it stands, is quite cryptic. Let's unwind it, to understand what it means in detail.

First of all, $\mathcal{I}X$ can be considered as a 2-category with only identity cells. The functor $\mathsf{K}_{\langle \mathcal{I},U\rangle X}$ is the constant functor mapping any g in $\mathcal{I}X$ to the functor $U_X : \mathcal{I}X \longrightarrow \mathsf{C}$, considered as an object of $\mathsf{Lax}_{\mathsf{C}}$. The functor $\langle \mathcal{I}, U\rangle \circ U_X$ maps an object g in $\mathcal{I}X$ to the functor $U_{U_X g} : \mathcal{I}U_X g \longrightarrow \mathsf{C}$, and a map $k: g \longrightarrow h$ to a 2-cell

$$\begin{tikzcd}
\mathcal{I}U_X g \arrow[rr, "\mathcal{I}U_X k"] \arrow[dr, "U_{U_X g}"'] \arrow[rr, phantom, shift right=4ex, "\Rightarrow j^{U_X k}"] & & \mathcal{I}U_X h \arrow[dl, "U_{U_X h}"] \\
& \mathsf{C}. &
\end{tikzcd} \tag{3.5}$$

Let's recall from [16, Definition 7.5.1 on p. 296] that a lax natural transformation α between parallel 2-functors F and G from C to D is a collection of maps $\alpha_X : FX \longrightarrow GX$ (X in C) such that, for any $f: X \longrightarrow Y$ in C, there is a 2-cell in D

$$\begin{tikzcd}
FX \arrow[r, "\alpha_X"] \arrow[d, "Ff"'] \arrow[dr, phantom, "\Rightarrow"] & GX \arrow[d, "Gf"] \\
FY \arrow[r, "\alpha_Y"'] & GY.
\end{tikzcd}$$

In our case, to give a lax natural transformation from $\langle\mathcal{I},U\rangle \circ U_X$ to $\mathsf{K}_{\langle\mathcal{I},U\rangle X}$ means, for each g in $\mathcal{I}X$, to give a functor $L(g)$ and a natural transformation g^L as below

$$\begin{tikzcd}
\mathcal{I}U_X g \arrow[rr, "L(g)"] \arrow[dr, "U_{U_X g}"'] \arrow[rr, phantom, shift right=4ex, "\Rightarrow g^L"] & & \mathcal{I}X \arrow[dl, "U_X"] \\
& \mathsf{C} &
\end{tikzcd} \tag{3.6}$$

and, for any $k\colon g \longrightarrow h$ in $\mathcal{I}X$, a transformation $k^L\colon Lg \Longrightarrow L(h)\mathcal{I}U_X k$ such that

$$U_X(k^L)g^L = (h^L \Diamond j^{U_X k})\colon U_{U_X g} \Longrightarrow U_X L(h)\mathcal{I}U_X k. \tag{3.7}$$

Diagrammatically, the two natural transformations are shown below.

$$\begin{tikzcd}
\mathcal{I}U_X g \arrow[r, "\mathcal{I}U_X k"] \arrow[dr, "U_{U_X g}"'] \arrow[dr, phantom, shift left=3ex, "\Rightarrow j^{U_X k}"] & \mathcal{I}U_X h \arrow[r, "Lh"] \arrow[d, "U_{U_X h}"] \arrow[dr, phantom, shift left=2ex, "\Rightarrow h^L"] & \mathcal{I}X \arrow[dl, "U_X"] \\
& \mathsf{C} &
\end{tikzcd}
\qquad
\begin{tikzcd}
& \mathcal{I}U_X g \arrow[dl, "U_{U_X g}"'] \arrow[d, "Lg"] \arrow[dr, "\mathcal{I}U_X k"] \arrow[dl, phantom, shift left=5ex, "\Rightarrow g^L"] \arrow[dr, phantom, shift right=5ex, "\Rightarrow k^L"] & \\
\mathsf{C} & \mathcal{I}X \arrow[l, "U_X"] & \mathcal{I}U_X h \arrow[l, "Lh"]
\end{tikzcd}$$

With this notion, we can now construct the substitution functions.

Assumption 4: The functor $\langle\mathcal{I},U\rangle$ has a lifting $\langle L, ()^L\rangle$.

Assumption 5: For any X in C_λ, the category $\mathcal{I}X$ is λ-filtered.

With these two further assumptions, for any map $f\colon X \longrightarrow \operatorname{colim} U_Y$ with X and Y in C_λ, we can derive the map $\mathsf{s}(f)\colon \operatorname{colim} U_X \longrightarrow \operatorname{colim} U_Y$ as follows.

Since X is λ-presentable and $\mathcal{I}Y$ is λ-filtered, there is a factorisation $f = \overline{i_f} \circ f^+$ where i_f is in $\mathcal{I}Y$:

$$\begin{tikzcd}
 & U_Y(i_f) \arrow[d, "\overline{i_f}"] \\
X \arrow[ur, "f^+"] \arrow[r, "f"'] & \operatorname{colim} U_Y
\end{tikzcd}$$

This defines a functor $\Lambda(f) = L(i_f)\mathcal{I}(f^+)$ and a natural transformation $f^\Lambda = i_f^L \Diamond j^{f^+} \colon U_X \Longrightarrow U_Y L(i_f)\mathcal{I}(f^+)$ as below.

$$\begin{tikzcd}
\mathcal{I}X \arrow[r, "\mathcal{I}(f^+)"] \arrow[rr, bend left, "\Lambda(f)"] \arrow[dr, bend right, "U_X"'] \arrow[dr, phantom, "\Rightarrow j^{f^+}"] & \mathcal{I}U_Y(i_f) \arrow[r, "L(i_f)"] \arrow[d, "U_{U_Y i_f}"] & \mathcal{I}Y \arrow[dl, bend left, "U_Y"] \arrow[dl, phantom, "\Rightarrow i_f^L"'] \\
 & \mathsf{C}. &
\end{tikzcd} \tag{3.8}$$

The natural transformation f^Λ, in turn, defines, by Lemma 3.9, the desired map

$$\mathrm{s}(f) = \operatorname{colim} f^\Lambda \colon \operatorname{colim} U_X \longrightarrow \operatorname{colim} U_Y. \tag{3.9}$$

The construction of $\mathrm{s}(f)$ appears to depend upon the factorisation $f = \overline{i_f} \circ f^+$. This is actually not the case.

Lemma 3.14 *Suppose $f = \overline{i_h} \circ h$ is another factorisation, inducing, as in (3.8), the functor $L(i_h)\mathcal{I}(h) \colon \mathcal{I}X \longrightarrow \mathcal{I}Y$ and the natural transformation $(i_h^L)_{\mathcal{I}(h)}.\mathsf{id} \colon U_X \Longrightarrow U_Y L(i_h)\mathcal{I}(h)$. Then, $\operatorname{colim}(i_f^L \Diamond j^{f^+}) = \operatorname{colim}(i_h^L \Diamond j^h)$ and $\mathrm{s}(f)$ is well-defined.*

Proof. Because $\mathcal{I}Y$ is λ-filtered, there exist in it an object i and arrows $k \colon i_f \longrightarrow i$, $k' \colon i_h \longrightarrow i$ such that f factorises as $f = \overline{i} \circ g$, for a given map

$g: X \longrightarrow U_Y(i)$, with $U_Y(k)f^+ = g = U_Y(k')h$, and the object i inducing a natural transformation $i^L \Diamond j^g : U_X \Longrightarrow U_Y L(i)\mathcal{I}(g)$ as in (3.6). Let's focus on $k: i_f \longrightarrow i$. By (3.7) and the third is the distribution of horizontal composition over vertical composition of natural transformations, one derives that

$$i^L \Diamond j^g = i^L \Diamond (j^{U_Y k} \Diamond j^{f^+}) = (U_Y(k^L) i_f^L) \Diamond j^{f^+} = (U_Y(k^L) \Diamond j^{f^+})(i_f^L \Diamond j^{f^+}).$$

Therefore, $U_Y(k^L) \Diamond j^{f^+}$ defines a 2-cell from $(i_f^L) \Diamond j^{f^+}$ to $(i^L) \Diamond j^g$, and, by Lemma 3.9, we have that

$$\operatorname{colim}(i^L) \Diamond j^g = \operatorname{colim}(i_f^L) \Diamond j^{f^+}.$$

By an analogue reasoning, one can show that $\operatorname{colim}(i^L \Diamond j^g) = \operatorname{colim}(i_h^L \Diamond j^h)$, thus proving, by transitivity, that the definition of $\mathrm{s}(f)$ does not depend on the chosen factorisation. □

Given any $f: X \longrightarrow TY$, where X and Y are finitely presentable, we shall henceforth denote the functor and natural transformation induced by the factorisation $f = \overline{i_f} \circ f^+$ by

$$\Lambda(f) = L(i_f)\mathcal{I}(f^+) \quad \text{and} \quad f^\Lambda = i_f^L \Diamond j^{f^+}. \tag{3.10}$$

3.2.3 Proving the Equations

In the previous section, we built, under suitable assumptions, the candidates for a Kleisli monoid on $[\mathsf{C}_\lambda, \mathsf{C}]$. We now turn to proving that equations 1–3 in Definition 3.10 hold.

Again, we will have to make some further assumptions. These may seem to overload the result, but they are satisfied by the examples we are trying to abstract from, so we think it's sensible to accept them.

Assumption 6: For each X in C_λ, $\operatorname{colim} i^L_X = \mathsf{id}_{TX}$.

If this holds, we can prove equation 1, i.e. that $\mathrm{s}(\eta_X) = 1_{UX}$, since $\mathrm{s}(\eta_X)$ can be constructed via a factorisation $\eta_X = \overline{i_X}\mathsf{id}_X$, thus

$$\mathrm{s}(\eta_X) = \operatorname{colim}(i^L_X)\Diamond j^{\mathsf{id}} = \operatorname{colim} i^L_X = \mathsf{id}_{TX}.$$

As for equation 2, i.e. $\mathrm{s}(f)\eta_X = f$, we need again some hypothesis.

Assumption 7: For any X and Y in C_λ, and for any $f\colon X \longrightarrow \operatorname{colim} U_Y$, there is a map $k\colon \Lambda(f)(i_X) \longrightarrow i_f$ in $\mathcal{I}Y$, such that $U_Y(k)f^\Lambda_{iX} = f^+$.

First of all, recall how $\mathrm{s}(f)$ is determined by a factorisation $f = \overline{i_f}.f^+$. Since η_X is the colimiting map from $U_X(i_X)$ to $\operatorname{colim} U_X$, $\mathrm{s}(f)\eta_X$ is the $U_X(i_X)$-th component of the cocone determined by f, that is $\mathrm{s}(f)\eta_X = \overline{\Lambda(f)(i_X)}f^\Lambda_{iX}$. Then, we have

$$\begin{tikzcd}
X = U_X(i_X) \arrow[r, "f^\Lambda_{iX}"] \arrow[dr, "f^+"'] & U_Y(\Lambda f(i_X)) \arrow[d, "U_Y k"'] \arrow[dr, "\overline{\Lambda(f)(i_X)}"] & \\
& U_Y(i_f) \arrow[r, "\overline{i_f}"'] & \operatorname{colim} U_Y
\end{tikzcd}$$

where the left triangle commutes by assumption 7 and the right since it is part of the universal cocone over U_Y.

Having verified two of the laws of a Kleisli monoid, we come to the last. For this, we need one last assumption on the lifting, which asserts some kind of functoriality over C.

Assumption 8: Given $f: X \longrightarrow TY$ and g in $\mathcal{I}X$, there is a 2-cell in any direction between the composites $\langle L(\Lambda(f)g), (\Lambda(f)g)^L\rangle\langle\mathcal{I}((f^\Lambda)_g), j^{(f^\Lambda)_g}\rangle$ and $\langle\Lambda(f), f^\Lambda\rangle\langle Lg, g^L\rangle$ in Lax_C.

Remark 3.15 More explicitly, Assumption 8 is saying that the two functors $\Lambda(f)Lg$ and $L(\Lambda(f)g)\mathcal{I}((f^\Lambda)_g)$ from $\mathcal{I}U_Xg$ to $\mathcal{I}Y$ are equal up to a 2-cell; i.e. there is a natural transformation in any direction filling the following square

$$\begin{tikzcd}
\mathcal{I}U_Xg \arrow[r, "Lg"] \arrow[d, "\mathcal{I}((f^\Lambda)_g)"'] \arrow[dr, phantom, "\Leftrightarrow"] & \mathcal{I}X \arrow[d, "\Lambda f"] \\
\mathcal{I}U_Y(\Lambda(f)g) \arrow[r, "L(\Lambda(f)g)"'] & \mathcal{I}Y,
\end{tikzcd} \tag{3.11}$$

and making the following equal via precomposition:

$$\begin{tikzcd}
\mathcal{I}U_Xg \arrow[r, "Lg"] \arrow[dr, bend right, "U_{U_Xg}"'] & \mathcal{I}X \arrow[r, "\Lambda f"] \arrow[d, "U_X"] & \mathcal{I}Y \arrow[dl, bend left, "U_Y"] \\
\arrow[r, phantom, "\Rightarrow_{g^L}"] & \mathsf{C} \arrow[r, phantom, "\Rightarrow_{f^\Lambda}"] &
\end{tikzcd}
\begin{tikzcd}
\mathcal{I}U_Xg \arrow[r, "\mathcal{I}((f^\Lambda)_g)"] \arrow[dr, bend right, "U_{U_Xg}"'] & \mathcal{I}U_Y(\Lambda(f)g) \arrow[r, "L(\Lambda(f)g)"] \arrow[d, "U_{U_Y(\Lambda(f)g)}"] & \mathcal{I}Y \arrow[dl, bend left, "U_Y"] \\
\arrow[r, phantom, "\Rightarrow_{j^{(f^\Lambda)_g}}"] & \mathsf{C} \arrow[r, phantom, "\Rightarrow_{(\Lambda(f)g)^L}"] &
\end{tikzcd} \tag{3.12}$$

All we have to prove for equation 3 is that, for λ-presentable objects X, Y and Z and maps $f: X \longrightarrow \operatorname{colim} U_Y$ and $g: Y \longrightarrow \operatorname{colim} U_Z$, we have $\mathrm{s}(\mathrm{s}(g)f) = \mathrm{s}(g)\mathrm{s}(f)$.

The map $\mathrm{s}(g)f$ can now be factorised as follows

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[dr, "f^+"'] & TY \arrow[r, "\mathrm{s}(g)"] & TZ \\
 & U_Yi_f \arrow[u, "\overline{i_f}"] \arrow[r, "(g^\Lambda)_{i_f}"'] & U_Z(\Lambda(g)i_f) \arrow[u, "\overline{\Lambda(g)i_f}"']
\end{tikzcd}$$

where $f = \overline{i_f}f^+$ is the factorisation of f used in the construction of $\mathrm{s}(f)$ and the square commutes by the construction of $\mathrm{s}(g)$. Thus, we have an

U_Z-factorisation of $\mathrm{s}(g)f$ and we calculate $\mathrm{s}(\mathrm{s}(g)f)$ as the colimit of $(\mathrm{s}(g)f)^\Lambda$, where, by (3.8), the functor $\Lambda(\mathrm{s}(g)f)$ can be calculated as

$$\Lambda(\mathrm{s}(g)f) = L(\Lambda(g)i_f)\mathcal{I}((g^\Lambda)_{i_f}f^+) = L(\Lambda(g)i_f)\mathcal{I}((g^\Lambda)_{i_f})\mathcal{I}(f^+) \qquad (3.13)$$

whereas

$$(\mathrm{s}(g)f)^\Lambda = (\Lambda(g)i_f)^L \Diamond j^{(g^\Lambda)_{i_f}} \Diamond j^{f^+}.$$

Analogously, we can express $\mathrm{s}(g)\mathrm{s}(f)$ as a colimit:

$$\mathrm{s}(g)\mathrm{s}(f) = \operatorname{colim}\langle\Lambda(g), g^\Lambda\rangle\operatorname{colim}\langle\Lambda(f), f^\Lambda\rangle = \operatorname{colim}\langle\Lambda(g)\Lambda(f), g^\Lambda \Diamond f^\Lambda\rangle.$$

Let's note that

$$\Lambda(g)\Lambda(f) = \Lambda(g)L(i_f)\mathcal{I}(f^+)$$

and

$$g^\Lambda \Diamond f^\Lambda = g^\Lambda \Diamond i_f^L \Diamond j^{f^+}.$$

Putting this together with (3.13) and assumption 8, we can derive a 2-cell between $\langle\Lambda(g)\Lambda(f), g^\Lambda \Diamond f^\Lambda\rangle$ and $\langle\Lambda(\mathrm{s}(g)f), (\mathrm{s}(g)f)^\Lambda\rangle$ (replace f by g and g by i_f in (3.11)); hence the two colimits are equal and the equation is proved.

Collating all the assumptions we made so far, we can finally state the main result of this chapter.

Theorem 3.16 *Let C be an $l\lambda p$ category, considered as a 2-category with just trivial 2-cells, $\langle\mathcal{I}, U\rangle\colon \mathsf{C} \longrightarrow \mathsf{Lax}_\mathsf{C}$ a 2-functor with a lifting $\langle L, ()^L\rangle$, such that $\mathcal{I}X$ is a λ-filtered category for any X in C_λ. Let for each X in C_λ be $TX = \operatorname{colim} U_X$, and assume that, for λ-presentable objects X and Y and for any $f\colon X \longrightarrow TY$, the following hold:*

1. *there is an object i_X in $\mathcal{I}X$ such that $U_X(i_X) = X$ and $\operatorname{colim} i_X^L = \mathsf{id}_{TX}$;*

2. *there is a map* $k\colon \Lambda(f)(i_X) \longrightarrow i_f$ *such that* $U_Y k f^{\Lambda}_{i_X} = f^+$;

3. *for all* g *in* $\mathcal{I}X$, *there is a 2-cell in any direction between the composites* $\langle \Lambda(f), f^{\Lambda}\rangle\langle Lg, g^L\rangle$ *and* $\langle L(\Lambda(f)g), (\Lambda(f)g)^L\rangle\langle \mathcal{I}((f^{\Lambda})_g), j^{(f^{\Lambda})_g}\rangle$ *in* $\mathsf{Lax}_{\mathsf{C}}$.

Then, the association $X \longmapsto TX$ *carries a Kleisli monoid structure.*

Corollary 3.17 *Under the assumptions of the previous theorem, the left Kan extension of the derived monoid* T *under the inclusion* $J\colon \mathsf{C}_\lambda \longrightarrow \mathsf{C}$ *is a finitary monad on* C.

Proof. By Proposition 3.11 and Theorem 3.16, T determines a functor, which by an abuse of notation we call again $T\colon \mathsf{C}_\lambda \longrightarrow \mathsf{C}$. This has a monoid structure, therefore its left Kan extension defines a monad on C, as explained in Remark 1.32. □

3.3 Deriving Guardedness

In previous sections, we worked out a result for building a monad as a pointwise colimit. The purpose, as already mentioned, is to pointwise get a set of terms over X for a signature Σ, and obtain a monad (thus modelling their closure under substitution) by taking their collection, i.e. a colimit. The monad T which we get out of this construction assigns to each X the free set of such terms over X. It is therefore reasonable to investigate whether T is F-guarded, or even strongly F-guarded, where F is the functor representing

Σ. More generally, of course, the question can be posed for any λ-accessible endofunctor of C.

So, let's fix an lλp category C and a functor F in $\mathsf{End}(\mathsf{C})_\lambda$. Let's also consider a 2-functor $\langle \mathcal{I}, U\rangle$ satisfying all the properties of Theorem 3.16. We want to find conditions on F and $\langle \mathcal{I}, U\rangle$ so that the monad T arising by Corollary 3.17 is F-guarded.

First of all, recall that the underlying functor T of T is obtained as a left Kan extension. We therefore need to transpose the notion of guardedness under the equivalence $\mathsf{End}(\mathsf{C})_\lambda \cong [\mathsf{C}_\lambda, \mathsf{C}]$.

Lemma 3.18 *Let F be a λ-accessible endofunctor on C and T a monoid in $\mathcal{M}on([\mathsf{C}_\lambda, \mathsf{C}])$. Suppose there is a natural transformation $\alpha\colon FT \longrightarrow T$ such that*

$$\begin{tikzcd}
F(T\otimes T) = FT\otimes T \arrow[r, "F\mu"] \arrow[d, "\alpha\otimes T"'] & FT \arrow[d, "\alpha"] \\
T\otimes T \arrow[r, "\mu"'] & T.
\end{tikzcd} \tag{3.14}$$

Then, $\mathsf{Lan}_J T$ is a λ-accessible F-guarded monad.

Vice versa, given any λ-accessible F guarded monad, precomposition with $J\colon \mathsf{C}_\lambda \longrightarrow \mathsf{C}$ induces a monoid T in $\mathcal{M}on([\mathsf{C}_\lambda, \mathsf{C}])$ and a natural transformation $\alpha\colon FT \longrightarrow T$ such that $\mu\alpha \otimes T = \alpha F\mu$.

Proof. It is trivial to see that application of Lan_J to diagram (3.14) gives the usual diagram expressing μ as an F-algebra homomorphism, since F, being λ-accessible, commutes with Lan_J. Likewise, precomposition of diagram (3.1) with J gives precisely diagram (3.14). □

However, the monoid structure on T is induced by a Kleisli monoid structure, so we want to express the notion equivalently for Kleisli monoids.

Lemma 3.19 *Let (T, η, s) be a Kleisli monoid in $[\mathsf{C}_\lambda, \mathsf{C}]$. Then, any assignation for all X in C_λ of a map $\alpha_X : FTX \longrightarrow TX$ such that for any $f: X \longrightarrow TY$ (X and Y λ-presentable) the following diagram commutes*

$$\begin{tikzcd}
FTX \arrow[r, "F\mathsf{s}(f)"] \arrow[d, "\alpha_X"'] & FTY \arrow[d, "\alpha_Y"] \\
TX \arrow[r, "\mathsf{s}(f)"'] & TY,
\end{tikzcd} \tag{3.15}$$

determines a natural transformation $\alpha: FT \longrightarrow T$ for which diagram (3.14) commutes, and vice versa.

Proof. Recall from the proof of Proposition 3.11 how the action of T on maps and the natural transformation μ are defined using s. $T(f)$, for a map $f: X \longrightarrow Y$ in C_λ, is defined as $\mathsf{s}(f\eta_Y)$, while $\mu_X : (\mathsf{Lan}_J T)TX \longrightarrow TX$ is determined by the cocone $\mathsf{s}(\overline{d}) : TX_d \longrightarrow TX$ where $\overline{d}: X_d \longrightarrow TX$ expresses TX as a colimit of λ-presentable objects and $(\mathsf{Lan}_J T)TX$ is the vertex of the colimiting cocone $\widetilde{d}: TX_d \longrightarrow (\mathsf{Lan}_J T)TX$.

The collection α_X clearly determines a natural transformation α, since, for a map $f: X \longrightarrow Y$ in C_λ, one has

$$T(f)\alpha_X = \mathsf{s}(f\eta_Y)\alpha_X = \alpha_Y F(\mathsf{s}(f\eta_Y)) = \alpha_Y FT(f).$$

To show that (3.14) commutes, note that $\widetilde{d}: TX_d \longrightarrow (\mathsf{Lan}_J T)TX$ is a λ-filtered colimit, hence it is preserved by F. Precomposing each component with $F\tilde{d}$, we then have the result by chasing the diagram below.

$$\begin{tikzcd}
FTX_d \arrow[dr, "F\tilde{d}"'] \arrow[drr, bend left, "Fs(\tilde{d})"] \arrow[ddd, "\alpha_{X_d}"'] & & \\
& F(\mathsf{Lan}_J T)TX \arrow[r, "F\mu_X"] \arrow[d, "(\mathsf{Lan}_J\alpha)_{TX}"'] & FTX \arrow[d, "\alpha_X"] \\
& (\mathsf{Lan}_J T)TX \arrow[r, "\mu_X"'] & TX \\
TX_d \arrow[ur, "\tilde{d}"] \arrow[urr, bend right, "\mathsf{s}(\tilde{d})"'] & &
\end{tikzcd}$$

Vice versa, since s is defined on a map $f\colon X \longrightarrow TY$ as the composite $\mu_Y T(f)$, one gets commutativity of (3.15) by pasting (3.14) with the naturality diagram for α on the map $T(f)$. □

So, in order to build an F-guarded monad by means of Theorem 3.16, we need to find conditions so that we get a collection of maps from FTX to TX, where T is the Kleisli monoid built therein.

Theorem 3.20 *Let* C*,* $\langle \mathcal{I}, U\rangle$ *and* $\langle L, ()^L\rangle$ *satisfy all the hypotheses of Theorem 3.16. Further, consider a* λ*-accessible endofunctor* F *on* C*. Suppose that, for all* X *in* C_λ*, there is a map* $\langle H_X, \alpha'_X\rangle$ *in* Lax_C *from* FU_X *to* U_X *such that for all* $f\colon X \longrightarrow TY$ *there is a 2-cell in* Lax_C *in any direction between* $\langle \Lambda(f)H_X, f^\Lambda \Diamond \alpha'_X\rangle$ *and* $\langle H_Y\Lambda(f), \alpha'_Y \Diamond F(f^\Lambda)\rangle$*:*

$$\begin{tikzcd}
\mathcal{I}X \arrow[rrr, "H_X"] \arrow[dr, "U_X"] \arrow[dd, "\Lambda(f)"'] & & & \mathcal{I}X \arrow[dl, "U_X"'] \arrow[dd, "\Lambda(f)"] \\
\arrow[r, phantom, "\Downarrow_{f^\Lambda}"] & \mathsf{Set} \arrow[r, "F"] \arrow[r, phantom, shift left=4ex, "\Rightarrow^{\alpha'_X}"] \arrow[r, phantom, shift right=4ex, "\Rightarrow^{\alpha'_Y}"] & \mathsf{Set} \arrow[r, phantom, "\Downarrow_{f^\Lambda}"] & \\
\mathcal{I}Y \arrow[rrr, "H_Y"'] \arrow[ur, "U_Y"'] & & & \mathcal{I}Y. \arrow[ul, "U_Y"]
\end{tikzcd} \tag{3.16}$$

Then, $\alpha_X = \operatorname{colim}\alpha'_X$ *determines a map from* FTX *to* TX *such that, for any* $f\colon X \longrightarrow TY$ *with* X *and* Y *in* C_λ*, (3.15) commutes.*

Proof. The map $\langle H_X, \alpha'_X\rangle$ consists of a natural transformation

$$\begin{tikzcd}
\mathcal{I}X \arrow[r, "H_X"] \arrow[d, "U_X"'] & \mathcal{I}X \arrow[d, "U_X"] \arrow[dl, phantom, "\Rightarrow \alpha'_X"] \\
\mathsf{C} \arrow[r, "F"'] & \mathsf{C},
\end{tikzcd}$$

inducing, by Lemma 3.9, a map $\alpha_X\colon \operatorname{colim} FU_X = F\operatorname{colim} U_X \longrightarrow \operatorname{colim} U_X$. Now, we want to prove that $\mathrm{s}(f)\alpha_X = \alpha_Y F\mathrm{s}(f)$. However, it follows from definition of s and α that $\mathrm{s}(f)\alpha_X = \operatorname{colim}(f^{\Lambda})\operatorname{colim}(\alpha'_X) = \operatorname{colim}(f^{\Lambda}\Diamond\alpha'_X)$, and, similarly, $\alpha_Y F\mathrm{s}(f) = \operatorname{colim}(\alpha'_Y\Diamond f^{\Lambda})$. The existence of a two cell in between the two maps, as in the statement, ensures that the two colimits are equal, therefore assuring commutativity of (3.15). □

Corollary 3.21 *Under the assumptions of the previous result, the left Kan extension of the monoid* T *is an* F*-guarded monad.*

In the examples provided in the next chapter, we shall also get some results on strong guardedness. The idea is that, analogously to what we did in Theorem 3.20, we pointwise get a coalgebra structure on TX by taking the colimit of a map in Lax_C from U_X to $(X+F)U_X$. Now, both the algebra and the coalgebra structure on TX are determined by some colimit of maps in Lax_C, therefore, we can ensure that these maps are inverse to each other by providing a 2-cell between the maps determining the composites and the identity on either U_X or $(X+F)U_X$.

3.4 An Application to Solution of Recursive Equations

A *recursive program scheme*, within a programming language, is a way of enriching the language itself by introducing new operators and equating them to known terms. The recursive path is created when the terms we are introducing are defined by means of themselves, as well as the already existing signature. The equations define the behaviour of a term, and, in order for them to be meaningful, i.e. to actually define a term, we need them to be productive.

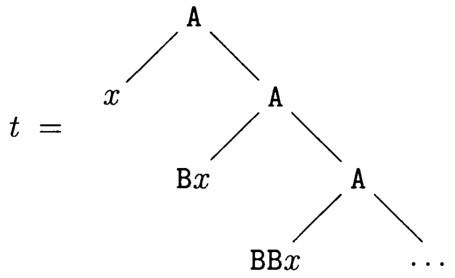
In a more algebraic fashion, a recursive program scheme is called an *algebraic system* [19].

If Σ is a signature, a Σ-*algebraic system of equations* is of the form

$$\begin{aligned} \phi_1(x_1, \ldots, x_{n_1}) &= t_1(x_1, \ldots, x_{n_1}) \\ \phi_2(x_1, \ldots, x_{n_2}) &= t_2(x_1, \ldots, x_{n_2}) \\ &\vdots \\ \phi_m(x_1, \ldots, x_{n_m}) &= t_m(x_1, \ldots, x_{n_m}) \end{aligned} \tag{3.17}$$

where the unknowns ϕ_i are the constructors of a signature Ω disjoint from Σ and the t_i's are finite terms built from the signature $\Sigma \cup \Omega$ over a countable set of variables X to which all the x's belong. The notion of productivity we mentioned above reflects in either the right handsides of the equations being variables from X or their root symbol being a Σ-constructor. Productive systems are called *guarded*, matching the notation of Section 1.1.3 and Section 2.4.

As an example, consider a signature Σ consisting of a binary symbol $\mathtt{A}$ and a unary symbol $\mathtt{B}$, and let Ω contain only a unary symbol ϕ. Then the Σ-algebraic equation $\phi(x) = \mathtt{A}(x, \phi(\mathtt{B}(x)))$ has the following solution



Reformulating these concepts in the categorical setting, we consider a signature $\Sigma\colon \mathcal{N}_\lambda \longrightarrow \mathsf{C}$ over an $l\lambda\mathrm{p}$ category and take the left Kan extension of Σ along the inclusion $JI\colon \mathcal{N}_\lambda \longrightarrow \mathsf{C}$; that is $F_\Sigma = \mathsf{Lan}_{JI}\Sigma$. We then take the free monad T_Σ over F_Σ. Within this terminology, a Σ-algebraic system of equations takes the form of a natural transformation from an unknown signature Ω to the set of terms $T_{\Sigma+\Omega}$. A term being guarded means that its root is a symbol from Σ. In other words, the term is a ground Σ-term with variables in the set of $\Sigma{+}\Omega$-terms, that is, an element of the set $F_\Sigma T_{\Sigma+\Omega} X$. A guarded Σ-algebraic system is then a natural transformation

$$\Omega \longrightarrow (\mathsf{Id} + F_\Sigma T_{\Sigma+\Omega}) \circ JI \tag{3.18}$$

where Ω is some signature. Note that, in doing this, we have allowed the mild generalisation in that the signature may declare an infinite number of operators.

Now, by abstract reasoning [37], $T_{\Sigma+\Omega} = T_\Sigma \oplus T_\Omega$ where the latter is the coproduct in the category of monads. It turns out that T_Ω can be replaced by any monad whatsoever, thus allowing us to consider algebraic equations where the right handside comes from any term algebra we choose. For T_Σ however, we cannot take any monad since the elements of the generalisation of T_Σ must be interpreted in the monad ${T^\nu}_\Sigma$ of infinite terms. Strongly F-guarded monads provide the right framework to do this. As a result of this abstraction process, we can now define algebraic equations as follows.

Definition 3.22 Let H be a strongly F-guarded monad. An *algebraic system* over H consists of a monad E and a monad morphism $e\colon E \longrightarrow \mathsf{Id} + F(H{\oplus}E)$, where $H{\oplus}E$ is the underlying functor of the coproduct of H and E in $\mathsf{Mon(C)}$, which is assumed to exist. A *solution* for e is a monad morphism $e^\dagger\colon \mathsf{E} \longrightarrow \mathsf{T}^\nu_F$ making the following commute ($!_H$ is the unique monad morphism from H to the final F-coalgebraic monad T^ν):

$$\begin{tikzcd}
E \arrow[r, "e^\dagger"] \arrow[d, "e"'] & T^\nu \arrow[d, "\cong"] \\
\mathsf{Id} + F(H{\oplus}E) \arrow[r, "\mathsf{Id}+F[!_H{,}e^\dagger]"'] & \mathsf{Id} + FT^\nu.
\end{tikzcd} \tag{3.19}$$

The very abstract and general formulation which was made possible by introducing guarded monads allows us to give a very nice and clean proof of a solution theorem for such equations in very different contexts, with the only assumption of our base category being locally presentable, and the functor F being accessible.

The proof of the solution theorem relies on the following lemma.

Lemma 3.23 *Let, for any X in C, $T^\nu X$ be the final $X + F$-coalgebra, thus determining the final strongly F-guarded monad T^ν as in Proposition 3.7. Let $\mathsf{H} = (H, \eta, \mu, \tau)$ be an F-guarded monad, and suppose that there is a monad* *morphism $\gamma\colon H \longrightarrow \mathsf{Id}+FH$ such that $\gamma.\tau = \mathsf{inr}.F\eta\colon F \longrightarrow \mathsf{Id}+FH$, making H a $(\mathsf{Id} + F \circ -)$-coalgebra. Then, the unique morphism σ to the final such coalgebra T^ν is also a monad morphism.*

$$\begin{tikzcd}
H \arrow[r, "\sigma"] \arrow[d, "\gamma"'] & T^\nu \arrow[d, "c"] \\
\mathsf{Id} + FH \arrow[r, "\mathsf{Id}+F\sigma"'] & \mathsf{Id} + FT^\nu
\end{tikzcd} \tag{3.20}$$

Proof. Let us call c the final coalgebra structure from T^ν to $\mathsf{Id}+FT^\nu$, which, because of Lambek's lemma, is an isomorphism, with inverse c^{-1}, as we saw in Proposition 3.7. Using the same notation as in Lemma 3.4, we can depict the situation in the following diagram

$$\begin{tikzcd}
 & H \arrow[r, "\sigma"] \arrow[d, shift left, "\gamma"] & T^\nu \arrow[d, shift left, "c"] \\
\mathsf{Id} \arrow[ur, bend left, "\eta"] \arrow[dr, "\mathsf{inl}"'] & & \\
 & \mathsf{Id} + FH \arrow[r, "\mathsf{Id}+F\sigma"'] \arrow[u, shift left, "\beta"] & \mathsf{Id} + FT^\nu, \arrow[u, shift left, "c^{-1}"]
\end{tikzcd} \tag{3.21}$$

where the two leftmost triangles commute, as well as the square involving γ. Let us write η^ν and μ^ν for the structure maps of the monad T^ν. Then, all we need to prove is that $\sigma\eta = \eta^\nu$ and $\sigma\mu = \sigma^2\mu^\nu$. The first equality follows because

$$\sigma\eta = \sigma\beta\mathsf{inl} = c^{-1}c\sigma\beta\mathsf{inl} = c^{-1}(\mathsf{Id} + F\sigma)\gamma\eta = c^{-1}(\mathsf{Id} + F\sigma)\mathsf{inl} = c^{-1}\mathsf{inl} = \eta^\nu.$$

A bit more work is needed in order to prove that σ respects multiplication. In order to achieve this, we will show that both $\sigma\mu$ and $\sigma^2\mu^\nu$ are coalgebra morphism from the same $(\mathsf{Id} + F \circ -)$-coalgebra into T^ν; finality of T^ν will then prove them equal. Here is the diagram relative to $\sigma\mu$.

$$\begin{tikzcd}
H^2 \arrow[rr, "\mu"] \arrow[d, "\gamma_H"'] & & H \arrow[r, "\sigma"] \arrow[dddd, "\gamma"'] & T^\nu \arrow[dddd, "c"] \\
(\mathrm{Id}+FH)H \arrow[d, "(\mathrm{Id}+FH)\gamma"'] & & & \\
\mathrm{Id}+FH+FH(\mathrm{Id}+FH) \arrow[d, "\mathrm{Id}+FH+FH\beta"'] & & & \\
\mathrm{Id}+FH+FH^2 \arrow[d, "\mathrm{Id}+[FH\eta,FH^2]"'] \arrow[rrd, "{[\mathrm{Id}+FH,F\mu]}"] & & & \\
\mathrm{Id}+FH^2 \arrow[rr, "\mathrm{Id}+F\mu"'] & & \mathrm{Id}+FH \arrow[r, "\mathrm{Id}+F\sigma"'] & \mathrm{Id}+FT^\nu
\end{tikzcd} \tag{3.22}$$

Here the commutativity of the top-left cell is just the fact that γ is a monad morphism.

The diagram for $\sigma^2\mu^\nu$ is slightly more complicated.

$$\begin{tikzcd}
H^2 \arrow[r, "H\sigma"] \arrow[d, "\gamma_H"'] & HT^\nu \arrow[r, "\sigma_{T^\nu}"] \arrow[d, "\gamma_{T^\nu}"'] & T^{\nu 2} \arrow[r, "\mu^\nu"] \arrow[d, "c_{T^\nu}"] & T^\nu \arrow[ddd, "c"] \\
H+FH^2 \arrow[r, "(\mathrm{Id}+FH)\sigma"] \arrow[d, "\gamma+FH^2"'] & T^\nu+FHT^\nu \arrow[r, "(\mathrm{Id}+F\sigma)_{T^\nu}"] \arrow[d, "c+F\sigma_{T^\nu}"'] & T^\nu+FT^{\nu 2} \arrow[d, "c+FT^{\nu 2}"] & \\
\mathrm{Id}+FH+FH^2 \arrow[r, "\mathrm{Id}+F\sigma+F\sigma^2"] \arrow[d, "\mathrm{Id}+[FH\eta,FH^2]"'] \arrow[rd, "{\mathrm{Id}+[FH\eta^\nu,FH\sigma]}"] & \mathrm{Id}+FT^\nu+FT^{\nu 2} \arrow[rd, "{\mathrm{Id}+[FT^\nu\eta^\nu,FT^{\nu 2}]}"] & \mathrm{Id}+FT^\nu+FT^{\nu 2} \arrow[d, "{\mathrm{Id}+[F\eta_{T^\nu},FT^{\nu 2}]}"] & \\
\mathrm{Id}+FH^2 \arrow[r, "\mathrm{Id}+FH\sigma"'] & \mathrm{Id}+FHT^\nu \arrow[r, "\mathrm{Id}+F\sigma_{T^\nu}"'] & \mathrm{Id}+FT^{\nu 2} \arrow[r, "\mathrm{Id}+F\mu^\nu"'] & \mathrm{Id}+FT^\nu
\end{tikzcd} \tag{3.23}$$

Unfortunately, the two coalgebras are not the same, since in the second case, after applying γ_H, one applies the identity on FH^2, whereas in the first case one applies $FH(\beta\gamma)$. This requires one more observation to close the proof. In diagram (3.23) the identity on FH^2 in the middle arrow of the left side is eventually followed by the map $FH\sigma$. Now, because of the universal property of σ, if we can show that $\beta\gamma$ is a $(\mathrm{Id}+F\circ -)$-coalgebra morphism, then we have that necessarily $\sigma = \sigma\beta\gamma$, because they are both the (unique) coalgebra homomorphism from H to T^{ν}, and we can safely substitute the coalgebra structure in (3.23) by the one in (3.22). In other words, we want to show that

$$\gamma\beta\gamma = (\mathsf{Id} + F\beta)(\mathsf{Id} + F\gamma)\gamma$$

which follows by commutativity of the following diagram, when precomposing the two outermost paths with γ:

$$\begin{tikzcd}
\mathsf{Id}+FH \arrow[rrr, "\mathsf{Id}+F\gamma"] \arrow[dd, "\beta"'] \arrow[dr, "\mathsf{Id}+\tau_H"] & & & \mathsf{Id}+F(\mathsf{Id}+FH) \arrow[dl, "{[\overline{\eta}\mathsf{inl},\mathsf{inr}F\eta_{(\mathsf{Id}+FH)}]}"'] \arrow[dd, "\mathsf{Id}+F\beta"] \\
& \mathsf{Id}+H^2 \arrow[r, "{[\overline{\eta}\mathsf{inl},\gamma^2]}"'] \arrow[dl, "{[\eta,\mu]}"'] & (\mathsf{Id}+FH)^2 \arrow[dr, "\overline{\mu}"] & \\
H \arrow[rrr, "\gamma"'] & & & \mathsf{Id}+FH.
\end{tikzcd}$$

Here, the left triangle is just the definition of β; the bottom square commutes because γ is a monad morphism; the right triangle commutes because of how $\overline{\mu}$ is defined, and, finally, the top square commutes because

$$(\mathsf{inr}F\eta)_{\mathsf{Id}+FH}(\mathsf{Id} + F\gamma) = (\gamma\tau)_{\mathsf{Id}+FH}F\gamma = \gamma_{\mathsf{Id}+FH}H(\gamma)\tau_H = \gamma^2\tau_H.$$

□

We can now state the solution theorem and give its proof.

Theorem 3.24 *If $e\colon E \longrightarrow 1 + F(H + E)$ is an algebraic system over a strongly F-guarded monad (H, η, μ, τ), then e has a unique solution.*

Proof. By composing τ with the injection of H into $H\oplus E$, one gets a natural transformation $\tau'\colon F \longrightarrow H\oplus E$, which, by Lemma 3.4, induces a monad structure on $1+F(H\oplus E)$. Moreover, the equality $\tau' = \mathsf{inl}\,\tau$ is satisfied by construction of τ', therefore, for the same lemma, there is also a monad morphism $1 + F\mathsf{inl}\colon 1 + FH \longrightarrow 1 + F(H\oplus E)$, which, being H coalgebraic, leads to a monad morphism $\delta\colon H \longrightarrow 1+F(H\oplus E)$ (just precompose $1+F\mathsf{inl}$ with the isomorphism between H and $1+FH$). Copairing δ with the equation morphism e gives a monad morphism $[\delta, e]\colon H\oplus E \longrightarrow 1 + F(H\oplus E)$, which endows $H\oplus E$ with a $(1 + F \circ -)$-coalgebra structure. We therefore have a coalgebra morphism to the final coalgebra $\beta\colon H\oplus E \longrightarrow T^{\nu}$, and we want to use Lemma 3.23 to show that this is a monad morphism. Precomposition with the second injection into the coproduct will then be our candidate solution morphism $e^{\dagger}\colon E \longrightarrow T^{\nu}$. In order to apply the lemma (where the H is now replaced by $H\oplus E$), we have to show that $[\gamma, e]\tau' = \mathsf{inr}F\eta'$, where η' is the unit of $H\oplus E$. But, since the coproduct is in the category of monads, one has that the left injection is a monad morphism, hence $\eta' = \mathsf{inl}\,\eta$. From this and the fact that H is coalgebraic, we get that

$$[\gamma, e]\tau' = [\gamma, e]\mathsf{inl}\tau = (1 + F\mathsf{inl})\mathsf{inr}'F\eta = \mathsf{inr}F(\mathsf{inl})F\eta = \mathsf{inr}F\eta',$$

where inr' is the injection of FH in $1 + FH$, inl maps H to $H\oplus E$, and inr is the inclusion of $F(H\oplus E)$ in $1 + F(H\oplus E)$.

All we have to do, now, is to show that diagram (3.19) commutes, but this trivially follows by looking at the second component of the commuting diagram (3.20), where H is now replaced by $H\oplus E$.

Now suppose d: $\mathsf{E} \longrightarrow \mathsf{T}^{\nu}$ is another solution of the algebraic system. By copairing it with the mediating morphism $!_H\colon H \longrightarrow T^{\nu}$, one gets a $(1+F\circ-)$-coalgebra morphism from $H\oplus E$ to T^{ν}, which is therefore the same morphism as γ. By precomposing with the right injection, now, one gets that $d = e^{\dagger}$, thus showing the uniqueness of the solution morphism. □

Chapter 4

Worked Examples

In the previous chapter we gave a theorem for generating monads whose functor part acts as a pointwise colimit. As mentioned there, the main reason for that theorem is to provide a unified construction for getting different monads of terms, and here we get two important examples.

Rational terms, defined in Section 1.1.3, are the first instance. Our work on them [26] was presented more or less when Adámek and his group first presented their construction of the free iterative monad [6]. At the time, they had more assumptions, and the result was somehow as difficult to achieve. In our case, we found it motivating enough to have a general way of producing strongly F-guarded monads, since different structures can be required in various areas of computing, and it is useful to have a unified way of tackling them.

In the second part of the chapter, instead, we shall focus on term graphs, and how to encode them as a monad.

4.1 Rational Terms

As we saw in Section 1.1.3, rational terms are the smallest set of terms closed under solution of finite systems of guarded equations. That is to say, given a guarded system

$$\begin{aligned} x_1 &= t_1 \\ &\vdots \\ x_n &= t_n \end{aligned}$$

where the terms t_i are rational terms with variables from the union $X \coprod Y$ of a set $X = \{x_1, \ldots, x_n\}$ of unknowns and a set Y of parameters, this admits precisely one solution, consisting of an n-tuple of rational terms.

In fact, more can be said, since any such system of equations is equivalent to a *flat* one.

Definition 4.1 A *flat system* of equations in Greibach normal form is a system of the form

$$x_i = t_i \qquad (i = 1, \ldots, n)$$

where $t_i \in F_\Sigma(\{x_1, \ldots, x_n\}) \coprod Y$.

In these systems, a variable x is equated to a constant, a parameter from Y or a ground term $f(x_1, \ldots, x_n)$, where x_i are variables (possibly including x itself) and f is an n-ary function symbol.

It is a known result (see for example [19]) that every finite system of guarded equations where the right hand-side is rational can be reduced to a finite flat system (the vice versa is obvious). In particular, every rational term appears as the solution of some flat system.

Example 4.2 The system consisting of just one equation

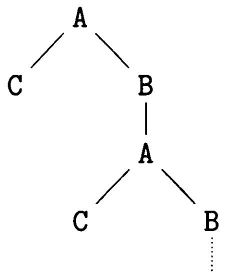
$$x = \mathtt{A}(\mathtt{C}, \mathtt{B}(x)) \tag{4.1}$$

reduces through a standard algorithm to the following flat system:

$$\begin{aligned} x_1 &= \mathtt{A}(x_2, x_3) \\ x_2 &= \mathtt{C} \\ x_3 &= \mathtt{B}(x_1) \end{aligned} \tag{4.2}$$

The solution to equation (4.1) is the regular tree

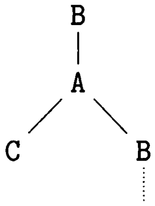
$t =$



(4.3)

whereas the solution of system (4.2) consists of the triple (t, u, v) where t is as above and u and v are its subtrees

$u = \mathtt{C}$ $v =$



By allowing the terms in the right hand-side to be variables from a set Y, we obtain parameterised solutions depending on Y.

The way one classically solves a flat system is to consider it as a map σ associating to each variable $x_i \in X$ the corresponding term in $Y + F_\Sigma X$, and by it determine a function $\widetilde{\sigma}$ on n-tuples of infinite terms, where n is the cardinality of X. The function $\widetilde{\sigma}$ is clearly contractive, because the system is guarded; moreover, $T^\nu(Y)^n$ is a complete metric space. Therefore, it admits a unique fix-point: an n-tuple $(t_1, \dots, t_n)$ such that $(t_1, \dots, t_n) = \widetilde{\sigma}(t_1, \dots, t_n)$.

The Functor $\langle \mathcal{I}, U \rangle$

As we already observed in Section 2.4, categorically, this argument is much better expressed by saying that σ is a $Y + F_\Sigma$-coalgebra. The solution is then given by the unique map ϕ to the final F_Σ-coalgebra, i.e. to $T^\nu Y$.

Rational terms are precisely those arising as solutions of such systems of equations. Therefore, in order to get an object of rational terms, we just have to "collect" all the solutions of the equations. That could be achieved by considering the solution morphisms of all equations and factorising them through their images (recall from [8] that all locally presentable categories have a regular epi-mono factorisation system). This would give us, for each Y in C, a collection of subobjects of $T^\nu Y$, and their union would be the object of rational terms over Y.

However, here we are going to choose a different approach, which does not rely on factorisation systems. In particular, we think of a system of equations as a formal way of presenting the terms which solve them. So, system (4.2) above "presents" the terms t, u and v shown thereafter.

In doing this association, of course, we have to be careful, since a rational term can be represented in many different ways (as there are many different equations of which it could be a solution). Think for example of the term

$$\mathtt{B}^{\omega} = \mathtt{B}(\mathtt{B}(\mathtt{B}(\ldots))).$$

This is clearly the solution of the equation $x = \mathtt{B}(x)$, as well as a solution of the system

$$\begin{aligned} x_1 &= \mathtt{B}(x_2) \\ x_2 &= \mathtt{B}(x_1) \end{aligned}$$

In order to get an object of rational terms, then, it is not enough to just collect all flat systems (by taking their coproduct); we also have to consider a quotient which identifies those equations having the same solution.

In the categorical model, flat systems are coalgebras with a finite carrier, and the solution is given by the unique map to the final coalgebra. Suppose $\gamma: Y \longrightarrow X + F_{\Sigma}(Y)$ and $\delta: Z \longrightarrow X + F_{\Sigma}(Z)$ are two such systems, with solution $\widetilde{\gamma}$ and $\widetilde{\delta}$ respectively. Then, any morphism $\phi: (Y, \gamma) \longrightarrow (Z, \delta)$ in $(X + F_{\Sigma})-\mathsf{Coalg}$ is such that $\widetilde{\delta}\phi = \widetilde{\gamma}$, i.e. it has to map equations from the first system to equations in the second which have the same solution.

The converse is also true. Given (Y, γ) and (Z, δ) as above, if there is a variable $y \in Y$ which resolves to the same term t as a variable $z \in Z$, then the smallest subcoalgebra (Y', θ) of (Y, γ) including y is isomorphic to the smallest subcoalgebra of (Z, δ) containing z. The two inclusions of (Y', θ) into (Y, γ) and (Z, δ) are coalgebra morphisms, and Y' is again finite.

This suggests that the object of rational terms with variables in X should be the collection of all $X + F_{\Sigma}$-coalgebras with a finite carrier, quotiented by the relation which identifies two states precisely when there is a coalgebra homomorphism mapping one to the other. That is exactly what happens if we take the colimit of the full subcategory of $(X + F_\Sigma) - \mathsf{Coalg}$ consisting of those coalgebras with a finite carrier (note that, since Set is locally small and there are, up to isomorphism, only a set of finite sets, no size issue arises when considering the colimit).

Of course, working in Set is here totally inessential, and we can consider in general any locally finitely presentable category C and any finitary endofunctor F on it. We shall consider the pair $\langle \mathcal{I}X, U_X \rangle$, where $\mathcal{I}X$ is the category of $X + F$-coalgebras with a finitely presentable carrier and coalgebra homomorphisms between them. This is a full subcategory of $X + F - \mathsf{Coalg}$, and U_X is defined as the restriction of the forgetful functor to C. Given a map $f: X \longrightarrow Y$ in C, its image along $\mathcal{I}$ is the functor $\mathcal{I}f: \mathcal{I}X \longrightarrow \mathcal{I}Y$ mapping an $X + F$-coalgebra (A, α) to the coalgebra

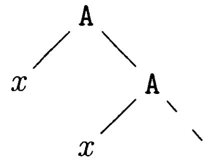
$$A \xrightarrow{\alpha} X + FA \xrightarrow{f+\mathsf{id}} Y + FA;$$

that is precisely variable renaming. Since $U_Y \mathcal{I}f = U_X$, we can define the natural transformation $j^f: U_X \Longrightarrow U_Y \mathcal{I}f$ as the identity.

Example 4.3 In order to give an idea of what will be going on during the calculations, in this section we shall present a few examples of rational terms, together with some coalgebras they are generated from in the colimit. We shall then read the operations on the coalgebras in terms of the results they produce on the corresponding trees, thus hoping to convey the intuition behind the laborious calculations of the previous chapter.

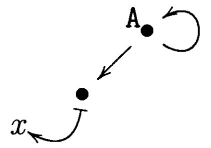
Our first example is that of the tree

$t =$



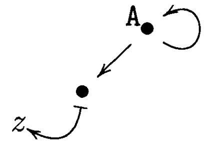
$\in R(\{x\})$.

Clearly, t is the image of y_1 along the colimiting map $\overline{\gamma}: Y \longrightarrow R(\{x\})$, where $Y = \{y_1, y_2\}$ and the coalgebra structure γ maps y_1 to $\mathtt{A}(y_2, y_1)$ and y_2 to x. Such a coalgebra we shall often represent pictorially as



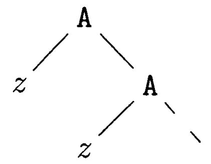
where the names of the states have been removed, the label of the transition performed by each state is written next to it, and the tagged arrows indicate that a particular state points to a variable, instead of performing an action. The term t is obtained by unfolding the node labelled by $\mathtt{A}$.

In this context, if $f: \{x\} \longrightarrow \{z\}$ is the obvious map, then $\mathcal{I}(f)(Y, \gamma)$ is the obvious coalgebra



where the variable x above has been renamed according to f . The unfolding of the node labelled by $\mathtt{A}$ gives the expected tree

$R(f)(t) =$



We shall define a monad of rational terms by mapping an object X to the colimit $RX = \operatorname{colim} U_X(\mathcal{I}X)$ in C. Note that R is trivially a functor, and it is finitary, since colimits and filtered colimits commute. Therefore, we know that $R = \mathsf{Lan}_J(RJ)$, and we can show that R is a monad by showing that its restriction to finitely presentable objects satisfies the hypotheses of Corollary 3.17.

Notice that, being C cocomplete, $X + F-\mathsf{Coalg}$ is too, and U_X creates colimits by Proposition 2.3. In particular, the carrier of a colimit of coalgebras is the colimit of the carriers. This, together with the fact that any finite colimit of finitely presentable objects is itself finitely presentable [8, Proposition 1.3], entails that $\mathcal{I}X$ is filtered.

The Lifting $\langle L, ()^L \rangle$

We now introduce a lifting for $\langle \mathcal{I}, U \rangle$. This is needed, as we saw in Section 3.2.2, to produce the substitution functions. The definition uses the same principle as in the proof of the Substitution Theorem in [1].

Given a coalgebra $\alpha\colon A \longrightarrow X + FA$ in $\mathcal{I}X$, we have that $\mathcal{I}(U_X\alpha)$ is the category of $A + F$-coalgebras. The lifting functor $L(\alpha)$ maps a coalgebra $\gamma\colon G \longrightarrow A + FG$ to the $X + F$-coalgebra

$$A+G \xrightarrow{[\mathsf{inl},\gamma]} A+FG \xrightarrow{\alpha+\mathsf{id}} X+FA+FG \xrightarrow{\mathsf{id}+[F\mathsf{inl},F\mathsf{inr}]} X+F(A+G). \tag{4.4}$$

Given another coalgebra $\xi\colon H \longrightarrow A + FH$ and a morphism between them $\phi\colon (G, \gamma) \longrightarrow (H, \xi)$, its image $L(\alpha)(\phi)$ is the coalgebra morphism

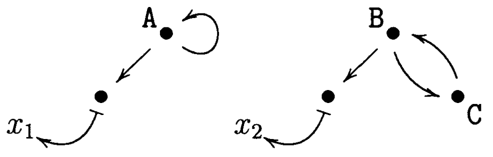
$$\mathsf{id} + \phi\colon (A + G, L\gamma) \longrightarrow (A + H, L\xi). \tag{4.5}$$

We also have to define the natural transformation $\alpha^L\colon U_A \Longrightarrow U_X L(\alpha)$. Its component on an $A + F$-coalgebra (G, γ) is the map $\mathsf{inr}\colon G \longrightarrow A + G$.

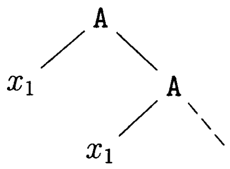
By (4.5), naturality is shown, on a morphism $\phi\colon (G,\gamma) \longrightarrow (H,\xi)$ by the chain of equalities in C $\alpha^L_\xi\phi = \mathsf{inr}\phi = (\mathsf{id}+\phi)\mathsf{inr} = L(\alpha)(\phi)\alpha^L_\gamma$.

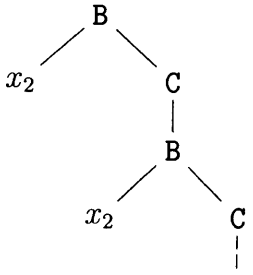
Example 4.4 The idea is that of starting with some terms with variables in X, which are represented by a coalgebra $\alpha\colon A \longrightarrow X + FA$. Given some terms with variables in A then (i.e. a coalgebra $\gamma\colon G \longrightarrow A + FG$), we substitute in them each variable from A with the term represented by that state in (A,α). The way this substitution is performed is by "putting together" the two coalgebras, creating some link which mimic for the variable states in (G,γ) the same behaviour as the states in (A,α) which they point to.

For example, let (A,α) be the coalgebra

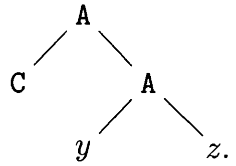


representing (amongst others) the trees

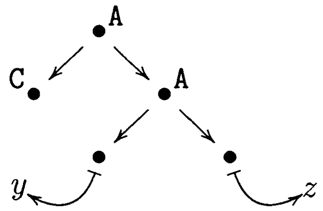
$t =$


and $u =$


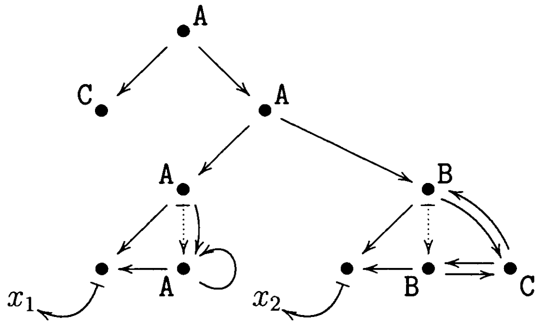
Let's suppose we want to perform the substitution $y \mapsto t$, $z \mapsto u$ in

$v =$


First of all, we represent v via a coalgebra (G, γ) depicted as

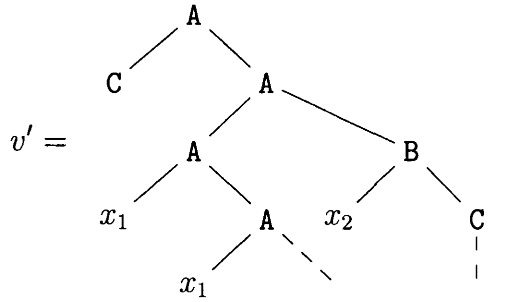


where v can be retrieved from the topmost node. The image of (G, γ) under $L(\alpha)$ is the coalgebra represented below.



Here, the dotted lines indicate the transitions determined by γ before the transformation, i.e. which particular elements of the carrier A the states were pointed to by γ (those are, in fact, those states whose unfolding in (A, α) gives rise to the terms t and u above). In $L(\alpha)(\gamma)$, those actions have been redefined in order to mimic exactly the ones of the variables which are being substituted for. In other words, the action of the image state along the dotted arrows has been *lifted* to the states in G. The dotted transitions are then removed (i.e. they are not part of the coalgebra $L(\alpha)(\gamma)$), therefore the states which they point "disappear" from the computation. In fact, we could have physically removed them, but in this case it is harmless to leave them in the coalgebra, because $\mathcal{I}X$ is a full subcategory of $X + F_\Sigma-\mathsf{Coalg}$, and (in Set) it contains all bisimulations between coalgebras with a finite carrier, therefore in the colimit all bisimilar states are identified and there is no reason to worry about redundant information. In Section 4.2 we shall have a much more refined description of $\mathcal{I}X$, and there we will have to remove inessential states.

If we now consider the unfolding of the topmost node, we obtain the tree



which is exactly the desired one: $v' = v[t/y, u/z]$.

Remark 4.5 Note that the map $\mathsf{inl}\colon A \longrightarrow A + G$ determines a morphism of $X + F$-coalgebras $\mathsf{inl}\colon (A, \alpha) \longrightarrow (A + G, L(\alpha)(G, \gamma))$, as easily checked by chasing the following diagram:

$$\begin{tikzcd}
A \arrow[r, "\mathsf{inl}"] \arrow[ddd, "\alpha"'] & A + G \arrow[d, "{[\mathsf{inl},\gamma]}"] \\
 & A + FG \arrow[d, "\alpha+\mathsf{id}_{FG}"] \\
 & X + FA + FG \arrow[d, "{\mathsf{id}_X+[F\mathsf{inl},F\mathsf{inr}]}"] \\
X + FA \arrow[r, "\mathsf{id}_X+F\mathsf{inl}"'] & X + F(A + G).
\end{tikzcd} \tag{4.6}$$

We can therefore construct, for any $f\colon X \longrightarrow RY$ factoring as $f = \overline{i_f} f^+$ for some $i_f\colon Y_0 \longrightarrow Y + FY_0$, the functor $\Lambda(f) = L(i_f) \circ \mathcal{I}(f^+)$ and the natural transformation f^Λ as in (3.10). In particular, given an $X + F$-coalgebra (G, γ), $\Lambda(f)(G, \gamma)$ is the coalgebra

$$Y_0{+}G \xrightarrow{[\mathsf{inl},(f^+{+}\mathsf{id})\gamma]} Y_0{+}FG \xrightarrow{i_f+\mathsf{id}} Y{+}FY_0{+}FG \xrightarrow{\mathsf{id}+[F\mathsf{inl},F\mathsf{inr}]} Y + F(Y_0{+}G)$$

and $f^{\Lambda}_{(G,\gamma)} = \mathsf{inr}\colon G \longrightarrow Y_0 + G$.

Notice that, because of Remark 4.5, we have that $\mathsf{inl}\colon Y_0 \longrightarrow Y_0 + G$ is a $Y+F$-coalgebra homomorphism from (Y_0, i_f) to $(Y_0+G, \Lambda(f)(G,\gamma))$ (replace A with Y_0, α with i_f and γ with $(f^{+}+\mathsf{id}_{FG})\gamma$ in (4.6)). In particular, it follows that

$$\overline{i_f} = \overline{\Lambda(f)(G,\gamma)}\mathsf{inl}. \tag{4.7}$$

The Properties

So, all we have to do in order to prove that R is a monad is to prove that assumptions 1 to 3 in Theorem 3.16 are satisfied by $\langle \mathcal{I}, U \rangle$.

- 1. If X is finitely presentable, then the coalgebra $i_X = \mathsf{inl}\colon X \longrightarrow X + FX$ is an object in $\mathcal{I}X$, and $U_X i_X = X$. In order to show that $\operatorname{colim} i^L_X = \mathsf{id}_{RX}$, we show that $\operatorname{colim} i^L_X = \operatorname{colim} \mathsf{Id}_{\mathcal{I}X}$, and this will follow, by Lemma 3.9, if we exhibit a 2-cell in $\mathsf{Lax}_{\mathsf{C}}$ between $\mathsf{Id}_{\mathcal{I}X}$ and $L(i_X)$, i.e. a natural transformation $\chi\colon \mathsf{Id}_{\mathcal{I}X} \longrightarrow L(i_X)$.

  The functor $L(i_X)$ maps an $X+F$-coalgebra (G,γ) to the $X+F$-coalgebra

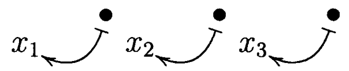
  $$X + G \xrightarrow{[\mathsf{inl},\gamma]} X + FG \xrightarrow{\mathsf{inl}+\mathsf{id}} X + FX + FG \xrightarrow{\mathsf{id}+[F\mathsf{inl},F\mathsf{inr}]} X + F(X+G)$$

  and the map $\mathsf{inr}\colon G \longrightarrow X + G$ is an $X+F$-coalgebra morphism from (G,γ) to $L(i_X)(G,\gamma)$, since

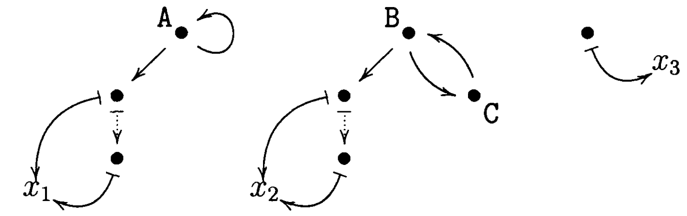
  $$\begin{aligned} L(i_X)(G,\gamma)\mathsf{inr} &= (\mathsf{id} + [F\mathsf{inl}, F\mathsf{inr}])(\mathsf{inl}+\mathsf{id})[\mathsf{inl},\gamma]\mathsf{inr} \\ &= (\mathsf{id} + [F\mathsf{inl}, F\mathsf{inr}])(\mathsf{inl}+\mathsf{id})\gamma \\ &= (\mathsf{id} + F\mathsf{inr})\gamma. \end{aligned}$$

We put $\chi_{(G,\gamma)} = \mathsf{inr}: G \longrightarrow X + G$. Naturality follows trivially, since, for a morphism $\phi: (G,\gamma) \longrightarrow (H,\xi)$, one has that $L(i_X)\phi = \mathsf{id}_X + \phi: X + G \longrightarrow X + H$.

Example 4.6 The coalgebra i_X can be represented pictorially as in Example 4.3 by a set consisting of as many nodes as elements in X, each pointing to a different element $x \in X$. So, if $X = \{x_1, x_2, x_3\}$, then the representation of i_X is



If we now consider the coalgebra (A,α) of Example 4.4, then $L(i_X)(\alpha)$ is the coalgebra



by which, unfolding the two topmost nodes on the left we get again the terms t and u of Example 4.4.

2. Given a map $f: X \longrightarrow TY$, where X and Y are finitely presentable objects, this factors as $f = \overline{i_f} f^+$ for some $Y+F$-coalgebra $i_f: Y_0 \longrightarrow Y + FY_0$, and $\Lambda(f)(i_x)$ is the coalgebra defined, following (3.10), as

$$Y_0+X \xrightarrow{\mathsf{inl}[\mathsf{id}_{Y_0}, f^+]} Y_0+FX \xrightarrow{i_f+\mathsf{id}} Y+FY_0+FX \xrightarrow{\mathsf{id}+[F\mathsf{inl}, F\mathsf{inr}]} Y+F(Y_0+X)\ .$$

We need to find a morphism $k: (Y_0 + X, \Lambda(f)(i_X)) \longrightarrow (Y_0, i_f)$ in $\mathcal{I}Y$ such that $U_Y(k)f^L_{i_X} = f^+$. We choose to this purpose the map $k =$

$[\mathsf{id}_{Y_0}, f^+]\colon Y_0 + X \longrightarrow Y_0$. This is a homomorphism, since

$$\begin{aligned}(\mathsf{id}_Y + Fk)\Lambda(f)(i_X) &= (\mathsf{id}_Y + [F\mathsf{id}_{Y_0}, Ff^+])(i_f + \mathsf{id}_{FX})\mathsf{inl}[\mathsf{id}_{Y_0}, f^+]\\ &= (\mathsf{id}_Y + F[\mathsf{id}_{Y_0}, f^+])i_f[\mathsf{id}_{Y_0}, f^+]\\ &= i_f[\mathsf{id}_{Y_0}, f^+]\\ &= i_f k.\end{aligned}$$

The i_x-th component of f^{Λ} is the map $\mathsf{inr}\colon X \longrightarrow Y_0 + X$, and it is clear that $U_Y(k)(f^{\Lambda})_{i_X} = [\mathsf{id}_{Y_0}, f^+]\mathsf{inr} = f^+$.

3. Let now (G, γ) be an object in $\mathcal{I}X$. We want to show that

$$\Lambda(f)L(g) = L(\Lambda(f)g)\mathcal{I}((f^{\Lambda})_g)\colon \mathcal{I}G \longrightarrow \mathcal{I}Y$$

and $f^{\Lambda}\Diamond g^L = (\Lambda(f)g)^L$ (recall that the natural transformation j is the identity).

Given a coalgebra $\alpha\colon A \longrightarrow G + FA$ in $\mathcal{I}G$, it is easy to compute that both $\Lambda(f)L(G)\alpha$ and $L(\Lambda(f)G)\alpha$ are the $Y + F$-coalgebra

$$\begin{tikzcd}
Y_0 + G + A \arrow[d, "{[\mathsf{inl},(f^+ + [F\mathsf{inl},F\mathsf{inr}])(g+\mathsf{id})[\mathsf{inl},\alpha]]}"] \\
Y_0 + F(G + A) \arrow[d, "i_f + \mathsf{id}"] \\
Y + FY_0 + F(G + A) \arrow[d, "{\mathsf{id}+[F\mathsf{inl},F\mathsf{inr}]}"] \\
Y + F(Y_0 + G + A)
\end{tikzcd}$$

and likewise, action on maps agrees.

The (A, α)-th component of the natural transformation $f^{\Lambda}\Diamond g^{L}$ is the map

$$A \xrightarrow{\mathsf{inr}} G + A \xrightarrow{\mathsf{inr}} Y_0 + G + A$$

which, by associativity of coproducts, is equal to the (A, α)-th component of $(\Lambda(f)g)^{L}$, i.e. the map $\mathsf{inr}: A \longrightarrow Y_0 + G + A$.

Guardedness

We can then apply Corollary 3.17, and conclude that R is a monad. We now want to show that it is also F-guarded; in fact, strongly guarded. In particular, we shall pointwise give, for each X in C_{fp}, an $X + F$-algebra structure ζ_X on RX. We shall prove ζ_X to be an isomorphism of the form $\zeta_X = [\eta_X, \alpha_X]$, with α_X satisfying the requirements of Lemma 3.19, hence the result. In the process, we shall need one further assumption, i.e. that F preserves finite presentability. That is to say that FX is finitely presentable whenever X is. This condition is met whenever C is Set and $F = F_{\Sigma}$ for some finite signature Σ.

First of all, let's define $\zeta_X: X + FRX \longrightarrow RX$. Since the functor $X + F$ is finitary, we have $X + FRX = (X + F)\mathrm{colim}\, U_X = \mathrm{colim}\,(X + F)U_X$, with cocone $X + F\overline{\gamma}: X + FG \longrightarrow X + FRX$, and we get an algebra map as a colimit if we can exhibit a map $\langle H_X, \zeta'_X \rangle$ in $\mathsf{Lax}_{\mathsf{C}}$ as follows:

$$\begin{tikzcd}
\mathcal{I}X \arrow[r, "H_X"] \arrow[d, "U_X"'] \arrow[dr, phantom, "\Rightarrow \zeta'_X"] & \mathcal{I}X \arrow[d, "U_X"] \\
\mathsf{C} \arrow[r, "X+F"'] & \mathsf{C}.
\end{tikzcd}$$

Analogously, we shall get the inverse $\beta_X \colon RX \longrightarrow X + RX$ of ζ_X by providing a map $\langle K_X, \beta'_X \rangle$ as below

$$\begin{tikzcd}
\mathcal{I}X \arrow[r, "K_X"] \arrow[d, "U_X"'] \arrow[dr, phantom, "\Rightarrow^{\beta'_X}"] & \mathcal{I}X \arrow[d, "U_X"] \\
\mathsf{C} & \mathsf{C}. \arrow[l, "X+F"]
\end{tikzcd}$$

In particular, we shall pose

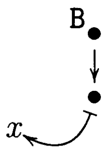
$$H_X(G,\gamma) = (X + FG, \mathsf{id}_X + F\gamma) \qquad (\zeta'_X)_{(G,\gamma)} = \mathsf{id}_{X+FG} \tag{4.8}$$

and

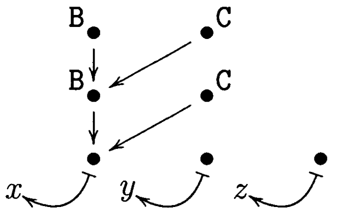
$$K_X = \mathsf{Id}_{\mathcal{I}X} \qquad (\beta'_X)_{(G,\gamma)} = \gamma \colon G \longrightarrow X + FG. \tag{4.9}$$

The corresponding maps will then be $\zeta_X = \operatorname{colim} \zeta'_X$ and $\beta_X = \operatorname{colim} \beta'_X$. Note that in order to define H_X it is essential to have that F preserves finite presentability, so that we can be sure that $X + FG$ is finitely presentable.

Example 4.7 The map ζ_X is meant to take a set of rational trees, form ground terms with variables over them, and "flatten" such terms giving back new rational terms. If (G, γ) is the coalgebra



representing the terms $t = \mathtt{B}(x)$ and $u = x$, $X = \{x, y, z\}$ and $\Sigma = \{\mathtt{B}, \mathtt{C}\}$ is a signature consisting of two unary symbols, then $H_X(G,\gamma)$ is the coalgebra



representing the terms x, y, z, $\mathtt{BB}x$, $\mathtt{B}x$, $\mathtt{CB}x$ and $\mathtt{C}x$, as expected.

Conversely, the coalgebra map β_X takes a term to the set of subtrees originated by removing the root of the term itself. All such terms are represented within the coalgebra we start with, therefore there is no need to alterate it. For this reason $K_X = \mathsf{Id}$.

In order to prove that $\beta_X\zeta_X = \mathsf{id}_{X+_FRX}$ and $\zeta_X\beta_X = \mathsf{id}_{RX}$, it is enough, by Lemma 3.9, to give a 2-cell $\sigma\colon \langle \mathsf{Id}, \mathsf{id}\rangle \longrightarrow \langle K_XH_X, \beta'_X\Diamond\zeta'_X\rangle$ and another $\tau\colon \langle \mathsf{Id}, \mathsf{id}\rangle \longrightarrow \langle H_XK_X, \zeta'_X\Diamond\beta'_X\rangle$ such that $U_X\sigma = \beta'_X\Diamond\zeta'_X$ and $(X+F)U_X\tau = \zeta'_X\Diamond\beta'_X$. It turns out that it is enough to consider $\sigma_{(G,\gamma)} = \tau_{(G,\gamma)} = \gamma$.

We now have to show that ζ_X is of the form $[\eta_X, \alpha_X]$, i.e. that $\zeta_X\mathsf{inl} = \eta_X$. We know, using the definition of ζ_X as a colimit, that

$$\begin{aligned}\zeta_X &= \zeta_X(X + F\overline{\gamma})\mathsf{inl}\\ &= \overline{X + F\gamma}(\zeta'_X)_{(G,\gamma)}\mathsf{inl}\\ &= \overline{X + F\gamma}\mathsf{inl},\end{aligned}$$

therefore we have that $\zeta_X\mathsf{inl} = \eta_X = \overline{i_X}$ if $\mathsf{inl}\colon X \longrightarrow X + FG$ is an $X + F$-coalgebra homomorphism from (X, i_X) to $(X + FG, X + F\gamma)$ and this is the case, since the following square commutes:

$$\begin{tikzcd}
X \arrow[r, "\mathsf{inl}"] \arrow[d, "\mathsf{inl}"'] & X + FG \arrow[d, "X+F\gamma"] \\
X + FX \arrow[r, "X+F\mathsf{inl}"'] & X + F(X + FG).
\end{tikzcd}$$

Finally, we should prove that, given a map $f\colon X \longrightarrow RY$, the square

$$\begin{tikzcd}
FRX \arrow[r, "F\mathsf{s}(f)"] \arrow[d, "\alpha_X"'] & FRY \arrow[d, "\alpha_Y"] \\
RX \arrow[r, "\mathsf{s}(f)"'] & RY
\end{tikzcd}$$

commutes, where $\alpha_X = \zeta_X \mathsf{inr}$, but we rather show that

$$\begin{tikzcd}
X + FRX \arrow[r, "k(f)"] \arrow[d, "\zeta_X"'] & Y + FRY \arrow[d, "\zeta_X"] \\
RX \arrow[r, "\mathsf{s}(f)"'] & RY
\end{tikzcd} \tag{4.10}$$

where $k(f)$ is the composite

$$X+FRX \xrightarrow{i_f f^+ + \mathsf{id}} Y+FY_0+FRX \xrightarrow{\mathsf{id}+F\mathsf{s}(f)} Y+FY_0+FRY \xrightarrow{Y+[F\overline{i_f},\mathsf{id}]} Y+FRY,$$

from which the desired result will follow by precomposing with the right injection $\mathsf{inr}\colon FRX \longrightarrow X + FRX$.

It is clear by (3.9) that $\mathsf{s}(f)\zeta_X = \operatorname{colim}(f^\Lambda \Diamond \zeta'_X)$. In order to show commutativity of (4.10), we can then use Lemma 3.9, provided we show that $k(f) = \operatorname{colim}(k')$ for some $\langle Q, k'\rangle\colon (X+F)U_X \longrightarrow (Y+F)U_Y$ and there is a natural transformation $\psi\colon \Lambda(f)H_X \longrightarrow H_Y Q$ with $U_Y(\psi)(f^\Lambda \Diamond \zeta'_X) = \zeta'_X \Diamond k'$, i.e. a 2-cell in $\mathsf{Lax}_{\mathsf{C}}$ between the following maps

$$\begin{tikzcd}
\mathcal{I}X \arrow[r, "H_X"] \arrow[d, "U_X"'] \arrow[dr, phantom, "\Rightarrow^{\zeta'_X}"] & \mathcal{I}X \arrow[r, "\Lambda(f)"] \arrow[dd, "U_X"'] \arrow[dr, phantom, "\Rightarrow^{f^\Lambda}"] & \mathcal{I}Y \arrow[ddl, bend left, "U_Y"] \\
\mathsf{C} \arrow[dr, bend right, "X+F"'] & & \\
& \mathsf{C} &
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathcal{I}X \arrow[r, "Q"] \arrow[d, "U_X"'] \arrow[dr, phantom, "\Rightarrow^{k'}"] & \mathcal{I}Y \arrow[r, "H_Y"] \arrow[d, "U_Y"] \arrow[dr, phantom, "\Rightarrow^{\zeta'_X}"] & \mathcal{I}Y \arrow[ddl, bend left, "U_Y"] \\
\mathsf{C} \arrow[dr, bend right, "X+F"'] & \mathsf{C} \arrow[d, "Y+F"] & \\
& \mathsf{C}. &
\end{tikzcd}$$

Let $\langle Q, k' \rangle$ be defined by the following composite:

$$\begin{tikzcd}
\mathcal{I}X \arrow[r, "\mathsf{Id}"] \arrow[d, "U_X"'] \arrow[dr, phantom, "\Rightarrow^{\mathsf{id}}"] & \mathcal{I}X \arrow[r, "\Lambda(f)"] \arrow[d, "U_X"] & \mathcal{I}Y \arrow[d, "U_Y"] \\
\mathsf{C} \arrow[r, "\mathsf{Id}"] \arrow[dr, bend right, "X+F"'] \arrow[dr, phantom, "\overset{i_f f^+ + F}{\Longrightarrow}"] & \mathsf{C} \arrow[d, "Y+FY_0+F"] \arrow[r, phantom, "\overset{Y+[F\mathsf{inl}, Ff^\Lambda]}{\Longrightarrow}"] & \mathsf{C} \arrow[dl, bend left, "Y+F"] \\
& \mathsf{C}. &
\end{tikzcd}$$

It is trivial that colim$\,(i_f f^+ + F) = i_f f^+ + \mathsf{id}\colon X + FRX \longrightarrow Y+FY_0+FRX$, so all we have to show is that colim$\,(Y+[F\mathsf{inl}, F(f^\Lambda)])$ is the composite

$$Y + FY_0 + FRX \xrightarrow{\mathsf{id}_{Y+FY_0}+F\mathsf{s}(f)} Y + FY_0 + FRY \xrightarrow{\mathsf{id}_Y+[F\overline{i_f},\mathsf{id}]} Y + FRY.$$

This follows by precomposing with an arbitrary colimiting map $Y+FY_0+F\overline{\gamma}$ (for $\gamma\colon G \longrightarrow X + FG$) and chasing the diagram

$$\begin{tikzcd}
Y+FY_0+FG \arrow[r, "\mathsf{id}_{Y+FY_0}+Ff^\Lambda"] \arrow[d, "\mathsf{id}_{Y+FY_0}+F\overline{\gamma}"] & Y+FY_0+F(Y_0+G) \arrow[r, "\mathsf{id}_Y+[F\mathsf{inl},\mathsf{id}]"] \arrow[d, "\mathsf{id}_{Y+FY_0}+F\overline{\Lambda(f)\gamma}"] & Y+F(Y_0+G) \arrow[d, "Y+F\overline{\Lambda(f)\gamma}"] \\
Y+FY_0+FRX \arrow[r, "\mathsf{id}_{Y+FY_0}+F\mathsf{s}(f)"'] & Y+FY_0+FRY \arrow[r, "\mathsf{id}_Y+[F\overline{i_f},\mathsf{id}]"'] & Y+FRY
\end{tikzcd}$$

where the top row is precisely the (G,γ)-th component of $\mathsf{id}_Y + [F\mathsf{inl}, F(f^\Lambda)]$, the left square commutes by definition of $\mathrm{s}(f)$, as in (3.9), and the right one because $\mathsf{inl}\colon Y_0 \longrightarrow Y_0 + G$ is a morphism of coalgebras from (Y_0, i_f) to $\Lambda(f)(G,\gamma)$, as we saw in (4.7), therefore

$$F(\overline{\Lambda(f)(G,\gamma)}\mathsf{inl}) = F\overline{i_f}.$$

Having introduced the pair $\langle Q, k'\rangle$, all we need in order to show commutativity of (4.10) is the natural transformation $\psi\colon \Lambda(f)H_X \longrightarrow H_YQ = H_Y\Lambda(f)$ mentioned above. On an $X+F$-coalgebra (G,γ), this will be the $Y+F$-coalgebra morphism

$$Y_0+X+FG \xrightarrow{[\mathsf{id},f^+]+\mathsf{id}_{FG}} Y_0+FG \xrightarrow{i_f+FG} Y+FY_0+FG \xrightarrow{\mathsf{id}_Y+[F\mathsf{inl},F\mathsf{inr}]} Y+F(Y_0+G).$$

That this is actually a coalgebra homomorphism and a natural transformation is a trivial diagram chase, and on any $X+F$-coalgebra (G,γ), both natural transformations have the same component

$$X+FG \xrightarrow{i_f f^+ + \mathsf{id}_{FG}} Y + FY_0 + FG \xrightarrow{\mathsf{id}_Y + [F\mathsf{inl},F\mathsf{inr}]} Y + F(Y_0+G)$$

so we have that $U_Y(\psi)(f^\Lambda \Diamond \zeta'_X) = \zeta'_X \Diamond k'$ and we have proved the following.

Proposition 4.8 *Given a finitary endofunctor F on C, the association*

$$X \longmapsto \operatorname{colim} \mathcal{I}X = RX$$

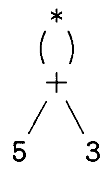
where $\mathcal{I}X$ is the category of $F+X$-coalgebras with a finitely presentable carrier, defines a monad R. If F preserves finite presentability, then R is strongly F-guarded.

The monad R is called the *rational monad* over F.

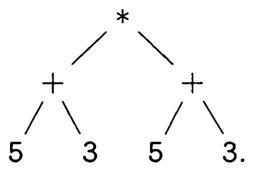
4.2 Term Graphs

In this section we shall work in the category Set, and the functor F will be of the form F_Σ for some finite and finitary signature Σ, unless otherwise explicitly said.

In this setting, we are going to apply Theorem 3.16 in order to give a categorical model of term graphs. These are widely used in computer science, for instance in implementing a functional programming language like Haskell. Intuitively, they are finite terms with loops and shared subterms. The advantage of representing programs as term graphs rather than as rational terms is that sharing variables reduces the number of computations, whereas the presence of loops still allows to perform recursion. For example, the program (5 + 3) ∗ (5 + 3) can be encoded as either

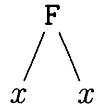


or

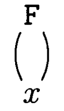


Clearly, in the first case the addition will be performed only once, whereas in the second case it will be evaluated twice.

When introducing term graphs, people often ask for variables to be shared to the maximum possible extent. Here, we choose to relax this condition, allowing such terms as



and



to coexist. This allows to choose for oneself how to handle the resources.

Concretely, we define term graphs over a Set-signature Σ as follows.

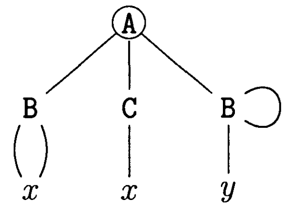
Definition 4.9 A *term graph* with variables ranging over a set X is a 6-tuple $(S, V, L_{\mathrm{s}}, L_{\mathrm{v}}, A, r)$ where

- S and V are finite sets;
- L_{s} is a function from S to Σ;
- $A: S \longrightarrow C^*$ is a function such that the length of the word $A(v)$ is equal to $\mathsf{ar}(L(v))$, where $C = S + V$ and C^* is the set of finite words over C;
- L_{v} is a function from V to X;
- r is an element of C such that for any other state s in C there is a finite sequence $a_1, \dots, a_n$ such that $a_1 = r$, $a_n = s$ and a_i is an element in the word $A(a_{i-1})$.

Here, C represents the set of *nodes* (or *states*, since later on they will be elements of the carrier of a coalgebra) of the term graph, naturally split as the union of those in S, which are *labelled* by a Σ-symbol according to L_{s}, and those in V, which point to some variable from X as specified by L_{v}. The function A maps each state in S to the (ordered) set of its *children* nodes, thought of as the arguments of the Σ-symbol specified by L_{s}. Clearly, there are as many children as the arity of the Σ-symbol. Finally, the element $r \in C$ is a chosen *root* of the term graph, from which all the other nodes can be reached.

For example, if $\Sigma = \{\mathtt{A}, \mathtt{B}, \mathtt{C}\}$ is a signature consisting of a ternary, a binary and a unary symbol respectively, then the term graph

$t =$



rooted at the circled node A, with variables in $X = \{x, y\}$, is represented by the 6-tuple $(S, V, L_{\mathrm{s}}, L_{\mathrm{v}}, A, r)$ where

$$\begin{array}{rcl@{\qquad}l@{\qquad}l@{\qquad}rcl}
S & = & \{s_1, s_2, s_3, s_4\} & L_{\mathrm{s}}(s_1) = \mathtt{A} & A(s_1) = s_2 s_3 & L_{\mathrm{v}}(v_1) & = & x \\
V & = & \{v_1, v_2, v_3\} & L_{\mathrm{s}}(s_2) = \mathtt{B} & A(s_2) = v_1 v_1 & L_{\mathrm{v}}(v_2) & = & x \\
r & = & s_1 & L_{\mathrm{s}}(s_3) = \mathtt{C} & A(s_3) = v_2 & L_{\mathrm{v}}(v_3) & = & y \\
 & & & L_{\mathrm{s}}(s_4) = \mathtt{B} & A(s_4) = v_3 s_4. & & &
\end{array}$$

These data can be presented much more elegantly as an $X + F_\Sigma$-coalgebra with carrier C. The structure map is

$$\gamma = [L_{\mathrm{v}}, \langle L_{\mathrm{s}}, A \rangle] : C = V + S \longrightarrow X + F_\Sigma(V + S) \tag{4.11}$$

where i is the inclusion of V in X as a subset, and $\langle L, A \rangle$ maps each state s to the pair $(L(s), A(s))$, which is an element of $F(C) = \coprod_{n \in \mathbb{N}} \Sigma_n \times C^n$ (more precisely, it will be an element of the particular $\Sigma_n \times C^n$ for which $n = \mathsf{ar}(L(s))$).

The information on the root transposes here to that of a generator.

Definition 4.10 Given an F-coalgebra (G, γ) for an endofunctor F on a category C, the *subcoalgebra of* (G, γ) *generated by* S , written $\langle S \rangle$, is the smallest subcoalgebra of (G, γ) containing S. Formally, if $i: S \longrightarrow G$ is the inclusion of S in G, then $\langle S \rangle = (H, \xi)$ is such that there is a monic $j\colon S \longrightarrow H$ and a coalgebra monomorphism $h\colon (H, \xi) \longrightarrow (G, \gamma)$ for which $U(h)j = i$, satisfying the universal property that for any other (H', ξ') with a monic $j'\colon S \longrightarrow H'$ and $h'\colon (H', \xi') \longrightarrow (G, \gamma)$ with $U(h')j' = i$ there is a (necessarily unique) morphism $k\colon (H, \xi) \longrightarrow (H', \xi')$ such that $h'k = h$ and $U(k)j = j'$.

In the particular situation we are dealing with now, the subcoalgebra generated by a subobject can be described explicitly in terms of reachable states. If (G, γ) is a coalgebra for the Set-endofunctor F_Σ and $S \subset G$, then the coalgebra $\langle S \rangle$ has as a carrier the set of all those states in G which can be reached starting from a state in S, in the sense made precise below.

First of all, let's observe that, since all polynomial Set-functors preserve weak pullbacks, the carrier of $\langle S \rangle$ is obtained by considering the intersection of the carriers of all subcoalgebras of (G, γ) including S [52]. In order to show the connections with the root of a term graph, though, we need to introduce the notion of path in a coalgebra.

Definition 4.11 Given an F-coalgebra (G, γ) and states $s, t \in G$, a *path* from s to t is a sequence $a_0, a_1, \dots, a_n$ in G with $a_0 = s$, $a_n = t$ and such that $a_{i+1} \in \gamma(a_i)$. Given a subset $S \subset G$ and a state t, we say that there is a path from S to t if there is a path from some $s \in S$ to t. If such a path exists, then we say that t is *reachable* from s (or S, accordingly).

When we say that $a_{i+1} \in \gamma(a_i)$ we are, in fact, abusing notations. We know that $\gamma(a_i)$ is of the form $\gamma(a_i) = (f, (g_1, \dots, g_m)) \in F(G)$, where f is an m-ary function from Σ. By $a_{i+1} \in \gamma(a_i)$ we mean that a_{i+1} is an element of the m-tuple $(g_1, \ldots, g_m)$.

Proposition 4.12 *Given an F-coalgebra (G, γ) and a subset S of G, the coalgebra $\langle S \rangle$ generated by S has as carrier the set $R(S)$ of all states in G reachable from S, with the obvious induced coalgebra structure.*

Proof. The set $R(S)$ is formally defined as

$$R(S) = \{g \in G \mid \exists a_0, \ldots, a_n \in G, a_0 \in S, a_n = g, a_{i+1} \in \gamma(a_i)\}.$$

To show that γ induces a coalgebra structure γ_S on $R(S)$, we need to show that for all g in $R(S)$ their image $\gamma(g)$ belongs to $F(R(S))$. If $a_0, \ldots, a_n$ is a path from $s \in S$ to g and $\gamma(g) = (f, (g_1, \ldots, g_m))$, then $a_0, \ldots, a_n, g_j$ is a path from s to g_j for all $j \in \{1, \ldots, m\}$, therefore $g_j \in R(S)$ and $\gamma(g) \in F(R(S))$. So, $(R(S), \gamma_S)$ is a subcoalgebra of (G, γ) including S. By the universal property characterising it, then, we have that $\langle S \rangle \subset R(S)$. We shall now prove that $R(S)$ is contained in any other subcoalgebra of (G, γ) containing S, so that $R(S) \subset \langle S \rangle$ and therefore they are equal. Let (G', γ') be a subcoalgebra of (G, γ) containing S, and let g be an element of $R(S)$. We shall prove, by induction on the length of a path from S to g, that g belongs to G'. If $g \in S$ (i.e. if the path has length 1), then trivially $g \in G'$. Suppose that $a_0, \ldots, a_n$ is a path from $a_0 \in S$ to $g = a_n$ and assume by inductive hypothesis that $a_0, \ldots, a_{n-1} \in G'$. Then, $g \in \gamma(a_{n-1}) = \gamma'(a_{n-1}) = (f, (g_1, \ldots, g_n))$, with $g = g_j$ for some j. Because (G', γ') is a subcoalgebra of (G, γ), we have that $g_1, \ldots, g_m \in G'$, and in particular $g \in G'$, hence the proof. □

Having clarified this, it is now obvious that the concept of root for a term graph has its categorical counterpart in the fact that (G, γ) is generated by r. Hence, the set of term graphs with variables from a set X is the set of $X + F$-coalgebras with a finite carrier, equipped with a choice of a specific generating element in the carrier (up to isomorphism). We shall call such a triple (g, G, γ) a *rooted coalgebra.*

Conversely, we can think of any rooted coalgebra as a term graph. Given (g, G, γ) with G finite, we can form the two pullbacks below:

$$\begin{tikzcd}
G_{\mathrm{v}} \arrow[r, "\mathsf{inl}"] \arrow[d, "\gamma_{\mathrm{v}}"'] & G \arrow[d, "\gamma"'] & G_{\mathrm{s}} \arrow[l, "\mathsf{inr}"'] \arrow[d, "\gamma_{\mathrm{s}}"] \\
X \arrow[r, "\mathsf{inl}"'] & X + FG & FG \arrow[l, "\mathsf{inr}"]
\end{tikzcd} \tag{4.12}$$

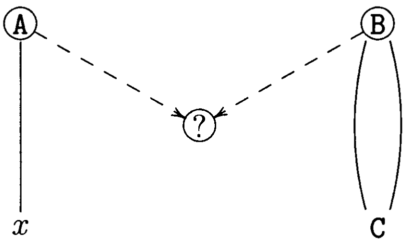
where clearly $G = G_{\mathrm{v}} + G_{\mathrm{s}}$. The map γ_{s} maps any element in G_{s} to a pair consisting of an n-ary term constructor and an n-tuple of states in G. We get the map L by composing γ_{s} with the first projection and the map A by composing it with the second. The map L_{v} is given by γ_{v}. The root, of course, will be given by g itself.

Remark 4.13 Note that, for the presence of roots, we can not abstract from Set to any lfp extensive category (where the same construction could otherwise be performed).

The Functor $\langle \mathcal{I}, U \rangle$

In order to apply Theorem 3.16 and get a monad of term graphs, we need to get $G(X)$ as a colimit. The most obvious way to achieve this would be to identify each term graph with its root, and consider their collection. It is not hard to see that this is in fact a set, because there are only a set of finite sets, up to isomorphism, and Set is locally small. So, the category $\mathcal{I}X$ would have as objects rooted $X+F$-coalgebras with a finite carrier. The forgetful functor U_X would map such a coalgebra to its root. Maps in $\mathcal{I}X$ would then have not to identify a root of a term graph to the root of a different one, otherwise the induced map via U_X would identify the two term graphs in the colimit. For this reason we would have to choose as maps in $\mathcal{I}X$ just isomorphisms of coalgebras (this way, we avoid having multiple copies of the same term graph), or equivalently, we could consider as objects representatives of the coalgebras up to isomorphism and the discrete category on such objects.

It is intuitively clear that the colimit of U_X as defined above is $G(X)$. However, this solution is not useful to our purpose, since for the theorem to work $\mathcal{I}X$ has to be filtered, and the category we just described is clearly not. For instance, there is no term graph to which we could map both of the following ones, because the roots (represented by the circled states) have different labels.



There is, however, an easy way around the problem. Let $\mathcal{I}'X$ be the category we just described. Then, the completion of $\mathcal{I}'X$ under finite coproducts in $X + F-\mathsf{Coalg}$ (in less categorical terms, we are considering *term graph* *forests*), is obtained as follows. Let C be the free category with finite coproducts over $\mathcal{I}'X$. Then, the inclusion of $\mathcal{I}'X$ into the cocomplete category $X+F-\mathsf{Coalg}$ determines a functor from C to $X+F-\mathsf{Coalg}$ which preserves finite coproducts. The completion of $\mathcal{I}'X$ in $X+F-\mathsf{Coalg}$ is the image of that functor. We shall take that image as our $\mathcal{I}X$. In order to show that it is a filtered category, it is now enough to show that C is, and this follows by the following lemma.

Lemma 4.14 *The free category with finite coproducts* D *over a discrete category* C *is filtered.*

Proof. Let C be a discrete category. Then, the elements of D have the form $D=\sum_{i=1}^{n} C_i$, where C_i is an object of C for any i. Arrows from such a D to $D'=\sum_{j=1}^{m} C'_j$ in D are of the form $f=[\mathsf{in}_{j_i}]_{i=1,\ldots,n}$, where for each i there is an index $j_i\in\{1,\ldots,m\}$ such that $C_i=C'_{j_i}$, in_{j_i} is the inclusion of C_i in the coproduct D' and f is the copairing of such injections.

We want to show that D is filtered. Given objects D and D' in D, we can clearly form their coproduct, and consider the inclusions

$$\begin{tikzcd} D \arrow[r, "\mathsf{inl}"] & D+D' & D' \arrow[l, "\mathsf{inr}"'] \end{tikzcd}$$

Given two parallel maps $f=[\mathsf{in}_{j_i}]_{i=1,\ldots,n}$ and $g=[\mathsf{in}_{l_i}]_{i=1,\ldots,n}$ from $D=\sum_{i=1}^{n} C_i$ to $D'=\sum_{j=1}^{m} C'_j$, we have that $C'_{j_i}=C'_{l_i}$ for all $i\in\{1,\ldots,n\}$, therefore we can consider the object

$$D''=\sum_{i=1}^{n} C'_{l_i}+\sum_{j\neq l_i, j\neq j_i} C'_j$$

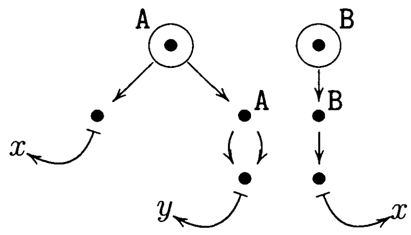
and the map $h\colon D'\longrightarrow D''$ defined as $h=[h_j]_{j=1,\ldots,n}$ where $h_j=\mathsf{in}_j$ if $j\neq l_j$, $j\neq j_i$ for all i, otherwise $h_j=\mathsf{in}_{l_i}\colon C'_{j_i}=C'_{l_i}\longrightarrow D''$. This clearly coequalises f and g, since it maps C'_{l_i} and C'_{j_i} to the same summand in D'', thus concluding the proof. □

So, we finally have a candidate for $\mathcal{I}X$. Objects are of the form $\Gamma = \sum_{i=1}^{n}(g_i, G_i, \gamma_i)$ where each (g_i, G_i, γ_i) is a rooted $X + F$-coalgebra with a finite carrier, and arrows are copairing of injections. Sometimes, we shall write a Γ as above in the form $\Gamma = (\{g_1, \ldots, g_n\}, G, \gamma)$, where $(G, \gamma) = \sum_{i=1}^{n}(G_i, \gamma_i)$ is a coproduct in $X + F-\mathsf{Coalg}$. The functor U_X will map such a Γ to the set $\{g_1, \ldots, g_n\}$.

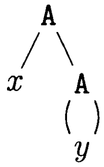
We know by Lemma 4.14 that $\mathcal{I}X$ is filtered, so all we have to do in order to apply Theorem 3.16 is to exhibit a lifting for $\langle \mathcal{I}, U \rangle$ and show that properties 1 to 3 therein are satisfied. Before that, though, we need to define the action of $\mathcal{I}$ on arrows. This is, indeed, quite straightforward: given a map $f: X \longrightarrow Y$ in Set, we associate to it the functor $\mathcal{I}f: \mathcal{I}X \longrightarrow \mathcal{I}Y$ mapping $(\{g_1, \ldots, g_n\}, G, \gamma)$ to $(\{g_1, \ldots, g_n\}, G, (f + \mathsf{id})\gamma)$, the natural transformation $j^f: U_X \longrightarrow U_Y\mathcal{I}(f)$ being the identity.

Example 4.15 As we did for the rational monad, we shall provide in this section a few examples, hoping to make the reader aware of the analogies and differences between the two cases.

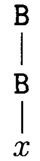
For example, in the case of the rational monad,the functor U was mapping a coalgebra to its carrier. In particular, an element x in the carrier was mapped to the rational term representing the evolution of the system starting at x. Here, instead, we map a term graph forest $\sum_{i=1}^{n}(g_i, G_i, \gamma_i)$ to the set $\{g_1, \ldots, g_n\}$, where g_i is meant to represent the whole term graph modelled by (G_i, γ_i). So, the coalgebra



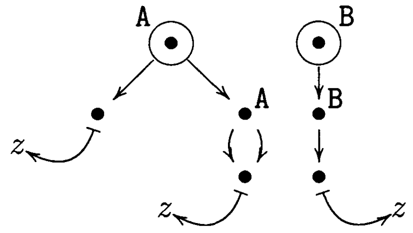
(where the roots are the circled states) will represent the term graphs



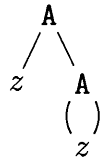
and



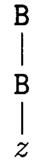
Functoriality of $\mathcal{I}$, again, models variable renaming. If $f\colon \{x, y\} \longrightarrow \{z\}$ is the unique possible arrow, for instance, then the coalgebra above becomes under the action of $\mathcal{I}(f)$



which models the term graphs



and



The Lifting $\langle L, ()^L \rangle$

We now turn to the definition of the lifting $\langle L, ()^L \rangle$. Given that $\mathcal{I}X$ is the free completion under finite coproducts of $\mathcal{I}'X$ in $X+F$–Coalg, we can define the functor $L(\Gamma)$ as the free extension of a functor from $\mathcal{I}'(\{g_1, \ldots, g_n\})$ to

$\mathcal{I}X$, where $\Gamma = (\{g_1, \ldots, g_n\}, G, \gamma)$ is an object of $\mathcal{I}X$. That is to say, the action of $L(\Gamma)$ is completely determined by the action on rooted coalgebras of the form (a, A, α) in $\mathcal{I}\{g_1, \ldots, g_n\}$. Such action we define as

$$L(\Gamma)(a, A, \alpha) = (\widetilde{a}, \widetilde{A}, \widetilde{\alpha})$$

where the triple $(\widetilde{a}, \widetilde{A}, \widetilde{\alpha})$ depends on whether a belongs to A_{s} or to A_{v}, defined by pullbacks as in (4.12).

If $a \in A_{\mathrm{s}}$, then we put $\widetilde{a} = a$,

$$\widetilde{A} = \sum_{a \in A_{\mathrm{v}}} G_a + A_{\mathrm{s}},$$

where for each $a \in A_{\mathrm{v}}$ $\alpha(a) = g_i$ for some i and G_a is the coalgebra G_i, and

$$\widetilde{\alpha} = \sum_{a \in A_{\mathrm{v}}} G_a + A_{\mathrm{s}} \xrightarrow{\sum \gamma_a + \alpha_{\mathrm{s}}} X + F(\sum_{a \in A_{\mathrm{v}}} G_a) + FA \xrightarrow{X + [F\mathsf{inl}, F(\widetilde{\alpha_{\mathrm{v}}} + \mathsf{id})]} X + F(\sum_{a \in A_{\mathrm{v}}} G_a + A_{\mathrm{s}}).$$

Here the function $\widetilde{\alpha_{\mathrm{v}}}\colon A_{\mathrm{v}} \longrightarrow \sum_{a \in A_{\mathrm{v}}} G_a$ maps an element a to the element $\alpha(a) = \alpha_{\mathrm{v}}(a)$ considered as an element of G_a.

If $a \in A_{\mathrm{v}}$, then it will be $L(\Gamma)(a, A, \alpha) = (g_a, G_a, \gamma_a)$ where the subscript a is the specific i for which $\alpha(a) = g_i$.

The component of the natural transformation Γ^L corresponding to the object $(\{a_1, \ldots, a_n\}, A, \alpha)$ has to be a map of the form

$$\{a_1, \ldots, a_n\} \longrightarrow \{g_{a_i} \mid a_i \in A_{\mathrm{v}}\} \cup \{a_i \mid a_i \in A_{\mathrm{s}}\}.$$

This will map the element a_i to itself if $a_i \in A_{\mathrm{s}}$, and to $g_{a_i} = \alpha(a_i)$ if $a_i \in A_{\mathrm{v}}$ (where $i = 1, \ldots, n$). Naturality is easily checked.

Finally, given a morphism $\phi\colon \Gamma \longrightarrow \Delta$ in $\mathcal{I}X$, we need a natural transformation $\phi^L\colon L(\Gamma) \longrightarrow L(\Delta)\mathcal{I}U_X\phi$ such that $U_X(\phi^L)\Gamma^L = \Delta\Diamond j^{U_X\phi}$.

Let's consider the object $A = \sum_{i=1}^{n}(a_i, A_i, \alpha_i)$ in $\mathcal{I}(U_X\Gamma)$. Then,

$$L(\Gamma)A = \sum_{i=1}^{n}(\widetilde{a_i}, \widetilde{A_i}, \widetilde{\alpha_i})$$

and writing ϕ_0 for $U_X\phi$

$$L(\Delta)\mathcal{I}(\phi_0)A = \sum_{i=1}^{n}(\widetilde{a_i'}, \widetilde{A_i'}, \widetilde{\alpha_i'}),$$

where $a_i' = a_i$, $A_i' = A_i$ and $\alpha_i' = (\phi_0 + FA_i)\alpha_i$. The coalgebra $(\widetilde{A_i'}, \widetilde{\alpha_i'})$ will therefore have as carrier the set $\sum_{a\in A_{i\mathrm{v}}} D_a + A_{i\mathrm{s}}$. Notice in particular that $A_{i\mathrm{v}}' = A_{i\mathrm{v}}$ and $\alpha_{i\mathrm{v}}' = \phi_0\alpha_{i\mathrm{v}}$, hence, because ϕ is a morphism in $\mathcal{I}X$, $(G_a, \gamma_a) = (D_a, \delta_a)$ and $g_{a_i} = d_{a_i}$ for all $i = 1, \ldots, n$.

In case $a_i \in A_{i\mathrm{s}}$, the two coalgebras $(\widetilde{a_i}, \widetilde{A_i}, \widetilde{\alpha_i})$ and $(\widetilde{a_i'}, \widetilde{A_i'}, \widetilde{\alpha_i'})$ become equal:

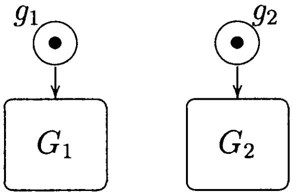
$$\begin{tikzcd}
\widetilde{A_i} = \sum\limits_{a\in A_{i\mathrm{v}}} G_a + A_{i\mathrm{s}} \arrow[d, swap, "\sum\limits_{a\in A_{i\mathrm{v}}} \gamma_a + \alpha_{i\mathrm{s}}"] & \sum\limits_{a\in A_{i\mathrm{v}}} D_a + A_{i\mathrm{s}} = \widetilde{A_i'} \arrow[d, "\sum\limits_{a\in A_{i\mathrm{v}}} \delta_a + \alpha_{i\mathrm{s}}"] \\
X + F(\sum\limits_{a\in A_{i\mathrm{v}}} G_a) + FA_i \arrow[d, swap, "\mathsf{id}+[F\mathsf{inl}, F(\widetilde{\alpha_{i\mathrm{v}}}+\mathsf{id}_{A_{i\mathrm{s}}})]"] & X + F(\sum\limits_{a\in A_{i\mathrm{v}}} D_a) + FA_i \arrow[d, "\mathsf{id}+[F\mathsf{inl}, F(\widetilde{\alpha_{i\mathrm{v}}'}+\mathsf{id}_{A_{i\mathrm{s}}})]"] \\
X + F(\sum\limits_{a\in A_{i\mathrm{v}}} G_a + A_{i\mathrm{s}}) & X + F(\sum\limits_{a\in A_{i\mathrm{v}}} D_a + A_{i\mathrm{s}}).
\end{tikzcd}$$

Likewise, if $a_i \in A_{i\mathrm{v}}$, then

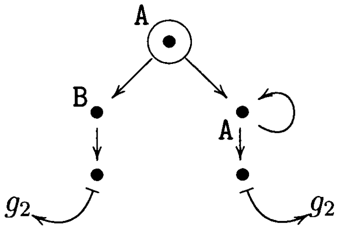
$$(\widetilde{a_i}, \widetilde{A_i}, \widetilde{\alpha_i}) = (g_{a_i}, G_{a_i}, \gamma_{a_i}) = (d_{a_i}, D_{a_i}, \delta_{a_i}) = (\widetilde{a_i'}, \widetilde{A_i'}, \widetilde{\alpha_i'}),$$

hence we can define ϕ^L to be the identity, which is clearly a morphism in $\mathcal{I}X$. The equation then reduces to $\Gamma^L = \Delta^L_{\mathcal{I}(U_X\phi)}$, and this is easily shown pointwise.

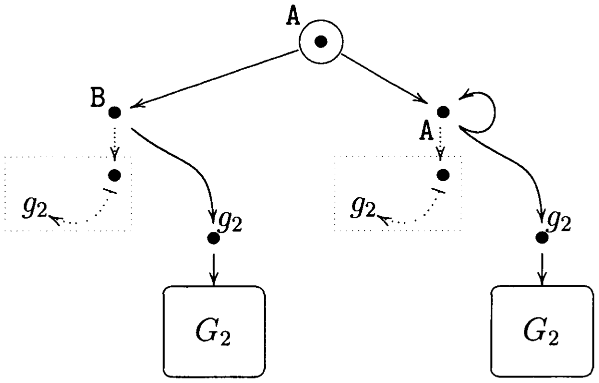
Example 4.16 The lifting is again at the core of the definition of the substitution functions. Suppose we have the coalgebra $\Gamma = (g_1, G_1, \gamma_1) + (g_2, G_2, \gamma_2)$. Here we are not interested in the internal structure of the G_i's, so we shall represent them diagrammatically by



where the circled g_i's are the roots, and the rest of each coalgebra G_i is enclosed in the corresponding box. Let's now consider the coalgebra

$(a, A, \alpha) =$ 

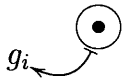
in $\mathcal{I}\{g_1, g_2\}$. Then, its image along $L(\Gamma)$ is obtained by removing each variable state and plugging in a copy of the subcoalgebra of Γ generated by the corresponding state:

$L(\Gamma)(a, A, \alpha) =$ 

Here the dotted parts have been removed, and the fresh copies of the corresponding coalgebra have been introduced.

In the case of the rational monad, we did not have to worry about removing redundant elements in the coalgebra. The reason for that was that, as explained in Example 4.4, the category $\mathcal{I}X$ there was including all bisimulations. Here, we explicitly avoided this to happen, by allowing only few arrows in $\mathcal{I}X$; therefore, our lifting is necessarily more refined.

In case $a \in A_{\mathrm{v}}$, then our coalgebra is necessarily of the form



for some i. Removing that only state and replacing it by the corresponding coalgebra as before yields the result (g_i, G_i, γ_i).

We are now in a position to define the substitution function s. Let $f\colon X \longrightarrow GY$ factor as

$$\begin{tikzcd}
X \arrow[rr, "f"] \arrow[dr, "f^+"'] & & GY \\
& Y_0 \arrow[ur, "i_f"'] &
\end{tikzcd} \tag{4.13}$$

where $Y_0 = \{y_1, \ldots, y_n\}$ and $i_f = \sum_{i=1}^{n}(y_i, Y_i, \chi_i)$.

Then, for $\Gamma = \sum_{j=1}^{m}(g_j, G_j, \gamma_j)$ in $\mathcal{I}X$, we have

$$\begin{aligned}
\Lambda(f)(\Gamma) &= L(i_f)\mathcal{I}(f^+)(\Gamma) \\
&= L(i_f)\sum_{j=1}^{m}(g_j, G_j, (f^+ + FG_j)\gamma_j) \\
&= \sum_{j=1}^{m}(\widetilde{g_j}, \widetilde{G_j}, \widetilde{\gamma'_j}),
\end{aligned} \tag{4.14}$$

where $\gamma'_j = (f^+ + FG_j)\gamma_j$, whereas the map

$$(f^{\Lambda})_{\Gamma} = (i_f^L \Diamond j^{f^+})\colon \{g_1, \ldots, g_m\} \longrightarrow \{f^+(g_j) \mid g_j \in G_{j_{\mathrm{v}}}\} \cup \{g_j \mid g_j \in G_{j_{\mathrm{s}}}\} \tag{4.15}$$

maps an element g_j to itself if $g_j \in G_{j_{\mathrm{s}}}$ and to $f^+(g_j)$ if $g_j \in G_{i\mathrm{v}}$.

The Properties

Bearing this in mind, we can now show that $\langle \mathcal{I}, U \rangle$ and $\langle L, ()^L \rangle$ satisfy properties 1–3 required in Theorem 3.16 to ensure that the functor G carries a monadic structure.

1. We need to exhibit an object i_X in $\mathcal{I}X$ such that $U_X i_X = X$ and $\operatorname{colim} i_X^L = \mathsf{id}_{GX}$. To this purpose, we shall put $i_X = (X, X, \mathsf{inl})$, which is clearly a coproduct:

$$i_X = \sum_{x \in X} (x, \{x\}, \mathsf{inl}) \quad \text{where} \quad \mathsf{inl}\colon \{x\} \longrightarrow \{x\} + F\{x\}.$$

Clearly, $U_X i_X = X$. In order to verify the other equation, consider the component $(i_X^L)_\Gamma = \sum_{i=1}^n (i_X^L)_{\Gamma_i}$ of i_X^L for an object $\Gamma = \sum_{i=1}^n (g_i, G_i, \gamma_i)$. For any i, we have the following morphism in $\mathcal{I}X$

$$\begin{tikzcd}
G_i \arrow[r, "\widetilde{\gamma_{i_\mathrm{v}}} + \mathsf{id}_{G_{i_\mathrm{s}}}"] \arrow[d, "\gamma_i"'] & \sum\limits_{g \in G_{i_\mathrm{v}}} \{x_g\} + G_{i\mathrm{s}} \arrow[d, "q + F(\widetilde{\gamma_{i_\mathrm{v}}} + \mathsf{id}_{G_{i_\mathrm{s}}})\gamma_{i_\mathrm{s}}"] \\
X + FG_i \arrow[r, "\mathsf{id}_X + F(\widetilde{\gamma_{i_\mathrm{v}}} + \mathsf{id}_{G_{i_\mathrm{s}}})"'] & X + F(\sum\limits_{g \in G_{i_\mathrm{v}}} \{x_g\} + G_{i\mathrm{s}})
\end{tikzcd}$$

where $q\colon \sum_{g \in G_{i_\mathrm{v}}} \{x_g\} \longrightarrow X$ maps x_g to itself as an element of X. The underlying map $U_X(\widetilde{\gamma_{i_\mathrm{v}}} + \mathsf{id}_{G_{i_\mathrm{s}}})$ is precisely the Γ_i-th component of i_X^L, whence

$$(i_X^L)_\Gamma = U_X(\sum_{i=1}^n (\widetilde{\gamma_{i_\mathrm{v}}} + \mathsf{id}_{G_{i_\mathrm{s}}})).$$

We then have that $\operatorname{colim}(i_X^L)$ is the only map making the following diagram commute for any Γ:

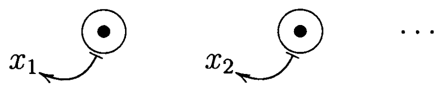
$$\begin{tikzcd}
U_X\Gamma \arrow[r, "(i^L_X)_\Gamma"] \arrow[d, "\overline{\Gamma}"'] & U_X(L(i_X)\Gamma) \arrow[d, "\overline{L(i_X)\Gamma}"] \\
GX \arrow[r, "\mathsf{colim}\,(i^L_X)"'] & GX;
\end{tikzcd}$$

but, since $(i^L_X)_\Gamma = U_X(\sum_{i=1}^n(\widetilde{\gamma_{i_\mathrm{v}}} + \mathsf{id}_{G_{i_\mathrm{s}}}))$, we have that

$$\overline{L(i_X)\Gamma}(i^L_X)_\Gamma = \overline{\Gamma},$$

hence $\mathsf{colim}\,(i^L_X) = \mathsf{id}_{GX}$.

Example 4.17 In particular, variables are still realised as in Example 4.6. Pictorially, we represent i_X as



where $X = \{x_1, \ldots, x_n\}$ is a finite set.

2. We require the existence of a map $k\colon \Lambda(f)(i_X) \longrightarrow i_f$ in $\mathcal{I}Y$ such that $U_Yk(f^\Lambda)_{i_X} = f^+$, with the same notation as in (4.13). The source of k has the form

$$\Lambda(f)(i_X) = \sum_{x\in X}(y_x, Y_x, \chi_x)$$

where $y_x = f^+(x) \in Y_0$, $Y_x = Y_i$ for that index such that $f^+(x) = y_i$ and $\chi_x = \chi_i\colon Y_i \longrightarrow Y+FY_i$ is a summand in i_f.

We shall denote by in^i_x the inclusion of the summand (y_x, Y_x, χ_x) in the coproduct $i_f = \sum_{i=1}^n(y_i, Y_i, \chi_i)$. Their copairing as x ranges over X will be denoted by $[\mathsf{in}^i_x]$. With this notation, we can define the map k

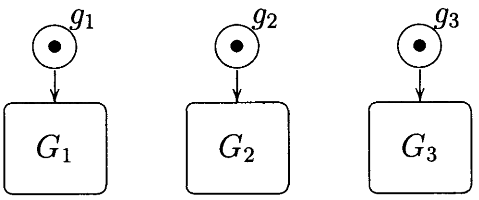
as follows:

$$\begin{tikzcd}
\sum_{x\in X}(y_x,Y_x,\chi_x) \arrow[r, "k={[\mathsf{in}^i_x]}"] \arrow[d, "\sum_{x\in X}\chi_x"'] & \sum_{i=1}^n(y_i,Y_i,\chi_i) = i_f \arrow[d, "\sum_{i=1}^n\chi_i"] \\
Y+F(\sum_{x\in X}(y_x,Y_x,\chi_x)) \arrow[r, "Y+F{[\mathsf{in}^i_x]}"'] & Y+F(\sum_{i=1}^n(y_i,Y_i,\chi_i)).
\end{tikzcd}$$

In particular, $k(y_x) = y_i = f^+(x)$ for all $x \in X$. It follows that $U_Y(k)(f^\Lambda)_{i_X}$ maps the element x to $f^+(x)$, therefore $U_Y(k)(f^\Lambda)_{i_X} = f^+$.

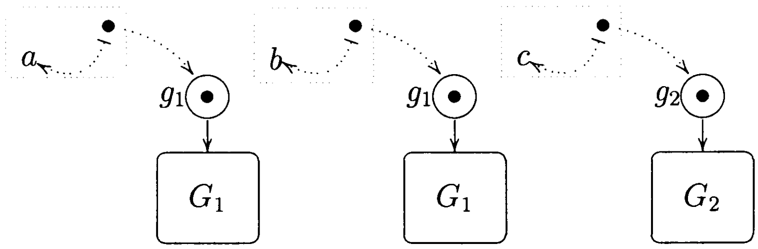
Example 4.18 Suppose $X = \{a,b,c\}$, and

$\mathcal{I}(Y) \ni i_f =$

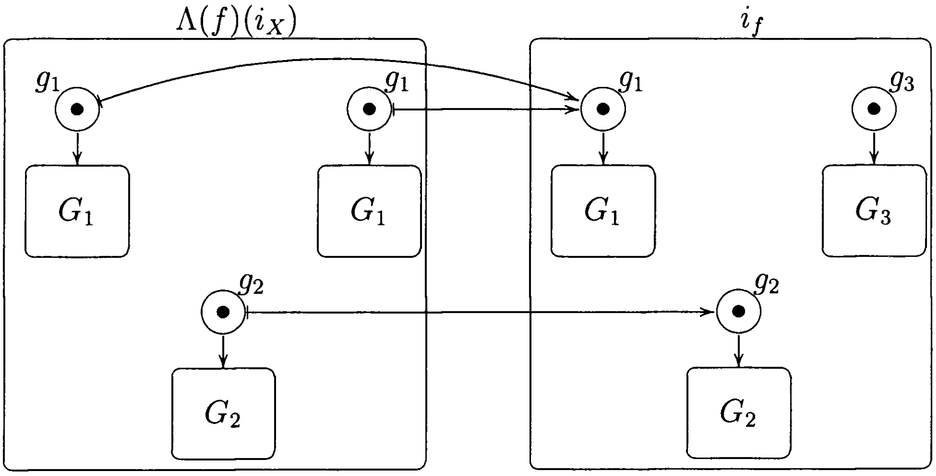


with $f^+\colon X \longrightarrow \{g_1,g_2,g_3\}$ mapping a and b to g_1 and c to g_2.

Then, $\Lambda(f)(i_X)$ is the coalgebra



where, as before, the dotted states have been removed. The map k will then act as follows



3. For this, let $A = \sum_{i=1}^{n}(a_i, A_i, \alpha_i)$ be an object in $\mathcal{I}(U_X\Gamma)$, where $\Gamma = \sum_{j=1}^{m}(g_j, G_j, \gamma_j)$ is an object in $\mathcal{I}X$, and $i_f = \sum_{l=1}^{p}(y_l, Y_l, \chi_l)$. Then, we want to define a 2-cell between the two following arrows in $\mathsf{Lax}_{\mathsf{Set}}$, where $\widetilde{\Gamma} = \Lambda(f)\Gamma$:

$$\begin{tikzcd}
\mathcal{I}\{g_1,\dots,g_n\} \arrow[r, "L\Gamma"] \arrow[rd, bend right, "U_{\{g_1,\dots,g_n\}}"'] \arrow[rd, phantom, "\Rightarrow\Gamma^L"] & \mathcal{I}X \arrow[r, "\Lambda f"] \arrow[d, "U_X"] \arrow[rd, phantom, "\Rightarrow f^{\Lambda}"] & \mathcal{I}Y \arrow[ld, bend left, "U_Y"] \\
& \mathsf{Set} &
\end{tikzcd}$$

and

$$\begin{tikzcd}
\mathcal{I}\{g_1,\dots,g_m\} \arrow[r, "\mathcal{I}((f^{\Lambda})_{\Gamma})"] \arrow[rd, bend right, "U_{\{g_1,\dots,g_n\}}"'] \arrow[rd, phantom, "\Rightarrow j^{(f^{\Lambda})_{\Gamma}}"] & \mathcal{I}\{\widetilde{g_1},\dots,\widetilde{g_m}\} \arrow[r, "L(\widetilde{\Gamma})"] \arrow[d, "U_{\{\widetilde{g_1},\dots,\widetilde{g_m}\}}"] \arrow[rd, phantom, "\Rightarrow(\widetilde{\Gamma})^L"] & \mathcal{I}Y \arrow[ld, bend left, "U_Y"] \\
& \mathsf{Set}. &
\end{tikzcd}$$

If we calculate the functor $\Lambda(f)L(\Gamma)$ on a rooted coalgebra (a, A, α)

with $a \in A_{\mathrm{s}}$, we get

$$\begin{aligned}
&\Lambda(f)L(\Gamma)(a, A, \alpha)\\
&= \Lambda(f)(a, \sum_{a \in A_{\mathrm{v}}} G_a + A_{\mathrm{s}}, [(\mathsf{id}_X + F\mathsf{inl}) \sum_{a \in A_{\mathrm{v}}} \gamma_a, F(\alpha_{\mathrm{v}} + \mathsf{id}_{A_{\mathrm{s}}})\alpha_{\mathrm{s}}])\\
&= L(i_f)(a, \sum_{a \in A_{\mathrm{v}}} G_a + A_{\mathrm{s}}, [(f^+ + F\mathsf{inl}) \sum_{a \in A_{\mathrm{v}}} \gamma_a, F(\alpha_{\mathrm{v}} + \mathsf{id}_{A_{\mathrm{s}}})\alpha_{\mathrm{s}}])\\
&= (a, (\sum_{a \in A_{\mathrm{v}}} (\sum_{g \in G_{a_{\mathrm{v}}}} Y_g) + G_{a_{\mathrm{s}}}) + A_{\mathrm{s}}, \xi)
\end{aligned}$$

where ξ is the composite

$$\begin{tikzcd}
(\sum_{a \in A_{\mathrm{v}}} (\sum_{g \in G_{a_{\mathrm{v}}}} Y_g) + G_{a_{\mathrm{s}}}) + A_{\mathrm{s}} \arrow[d, "(\sum_{a \in A_{\mathrm{v}}} (\sum_{g \in G_{a_{\mathrm{v}}}} \chi_g) + \gamma_{a_{\mathrm{s}}}) + \alpha_{\mathrm{s}}"'] \\
Y + \sum_{a \in A_{\mathrm{v}}} ((\sum_{g \in G_{a_{\mathrm{v}}}} FY_g) + FG_a) + FA \arrow[d, "\mathsf{id}_Y + [F\mathsf{inl}, F(\widetilde{\gamma_{a_{\mathrm{v}}}} + \mathsf{id}_{G_{a_{\mathrm{s}}}}), F(\widetilde{\alpha_{\mathrm{v}}} + \mathsf{id}_{A_{\mathrm{s}}})]"'] \\
Y + F((\sum_{a \in A_{\mathrm{v}}} (\sum_{g \in G_{a_{\mathrm{v}}}} Y_g) + G_{a_{\mathrm{s}}}) + A_{\mathrm{s}}).
\end{tikzcd} \tag{4.16}$$

In fact, there is a slight abuse of notation in the diagram above, whose purpose is to maintain the notation readable. When considering the coproduct of the coalgebras χ_g, we are generating several copies of Y, which are then all identified in the unique copy of Y appearing in the target of the first map.

The coalgebra $\Lambda(f)\Gamma$ has been described in (4.14), and with the same notation the set $\widetilde{G_j}$ now becomes the coproduct $\widetilde{G_j} = \sum_{g \in G_{j_{\mathrm{v}}}} Y_g + G_{j_{\mathrm{s}}}$, whereas γ_j' is the composite $(\mathsf{id}_Y + [F\mathsf{inl}, F(\widetilde{\gamma_{j_{\mathrm{v}}}} + \mathsf{id}_{G_{j_{\mathrm{s}}}})])(\sum_{g \in G_{j_{\mathrm{v}}}} \chi_g + \gamma_{i_{\mathrm{s}}})$. The map $(f^{\Lambda})_{\Gamma}$ was described in (4.15), and we shall write G_0 for its target. Using these elements, we can describe the action of the functor

$L(\Lambda(f)\Gamma)\mathcal{I}(f^\Lambda)_\Gamma$ on the same rooted coalgebra (a, A, α):

$$\begin{aligned} L(\Lambda(f)\Gamma)\mathcal{I}(f^\Lambda)_\Gamma(a, A, \alpha) &= L(\Lambda(f)\Gamma)(a, A, ((f^\Lambda)_\Gamma + \mathsf{id}_{FA})\alpha) \\ &= (a, \sum_{a \in A_\mathrm{v}} (\sum_{g \in G_{a\mathrm{v}}} Y_g + G_{a\mathrm{s}}) + A_\mathrm{s}, \beta) \end{aligned}$$

where $\sum_{g \in G_{a\mathrm{v}}} Y_g + G_{a\mathrm{s}}$ is the summand $\sum_{g \in G_{a\mathrm{v}}} Y_g + G_{a\mathrm{s}}$ for which $\alpha(a) = g_i$ and β is the same composite as in (4.16).

Analogously, if (a, A, α) is a rooted coalgebra with $a \in A_\mathrm{v}$, one can show that the two functors agree. Therefore, we have the equality

$$\Lambda(f)L(\Gamma) = L(\Lambda(f)\Gamma)\mathcal{I}(f^\Lambda)_\Gamma.$$

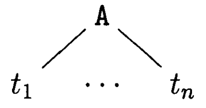
Moreover, the natural transformations $f^\Lambda \Diamond \Gamma^L$ and $(\widetilde{\Gamma})^L \Diamond j^{(f^\Lambda)_\Gamma}$ agree on each object, therefore we can choose the identity 2-cell between them.

With this, we have shown that the hypotheses of Theorem 3.16 are satisfied, proving the following:

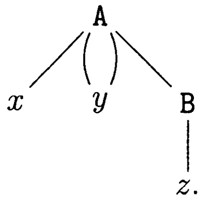
Proposition 4.19 *Given any finite and finitary signature Σ on* Set*, the monad mapping each set X to the set GX of Σ-term graphs over X defines a (finitary) monad.*

Guardedness

Let's now explore guardedness properties for the monad G we just described. Intuitively, it is clear that each set GX is an F-algebra, and also that this algebra preserves multiplication. However, there is no reason to expect such a structure to be bijective. In fact, the action will take an n-ary term constructor $\mathtt{A}$ and n term graphs $t_1, \ldots, t_n$ to give back the term graph



There is therefore no way to get a term whose root has a shared child, like the one depicted below; hence, the algebra structure is not surjective:



In order to prove formally that G is F-guarded, we shall use Theorem 3.20, or more precisely, its corollary.

For that, we have to exhibit for any set X, a map $\langle H_X, \alpha'_X \rangle$ in $\mathsf{Lax}_{\mathsf{Set}}$ from FU_X to U_X such that, for all $f\colon X \longrightarrow GY$, diagram (3.16) commutes up to a 2-cell.

Given an object $\Gamma = \sum_{i=1}^{n}(g_i, G_i, \gamma_i)$ in $\mathcal{I}X$, we put

$$H_X(\Gamma) = \sum_{t \in F\{g_1,\ldots,g_n\}} (t, H_t, \xi_t)$$

where (H_t, ξ_t) is defined as follows.

First of all, we look at the term $t \in F\{g_1, \ldots, g_n\}$; this will be a ground term over Σ, i.e. it will be of the form $\mathtt{A}(g_{i_1}, \ldots, g_{i_m})$ where m is the arity of $\mathtt{A}$ and $i_j \in \{1, \ldots, n\}$. The coalgebra (H_t, ξ_t) will then be defined as

$$\xi_t\colon H_t = \{t\} + \sum_{j=1}^{m} G_{i_j} \xrightarrow{[\vartheta_t, (\mathsf{id}_X + F\mathsf{inr}) \sum \gamma_{j_i}]} X + F(\{t\} + \sum_{j=1}^{m} G_{j_i})$$

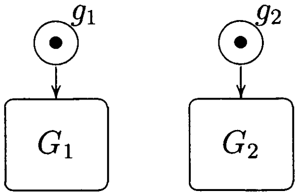
where $\vartheta_t\colon \{t\} \longrightarrow X + F(\{t\} + \sum_{j=1}^{m} G_{i_j})$ maps t to itself, considered as a term in $F\{g_1, \ldots, g_m\} \subset F(H_t)$.

Given another object $\Delta = \sum_{l=1}^{p}(d_l, D_l, \delta_l)$ and a morphism $\psi\colon \Gamma \longrightarrow \Delta$ in $\mathcal{I}X$, every g_i will be mapped to some d_{l_i}, with

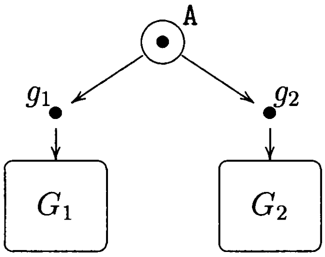
$$(G_i, \gamma_i) = (D_{l_i}, \delta_{l_i}). \tag{4.17}$$

Therefore, ψ induces a map $\psi'\colon \{g_1, \ldots, g_m\} \longrightarrow \{d_{l_1}, \ldots, d_{l_m}\}$. By applying F to the composite of ψ' with the injection of $\{d_{l_1}, \ldots, d_{l_m}\}$ in $\{d_1, \ldots, d_p\}$, we get a map $\widetilde{\psi}\colon F\{g_1, \ldots, g_m\} \longrightarrow F\{d_1, \ldots, d_p\}$. Notice that, because of (4.17), for each t in $F\{g_1, \ldots, g_m\}$, the coalgebra (H_t, ξ_t) is isomorphic to $(H_{\widetilde{\psi}(t)}, \xi_{\widetilde{\psi}(t)})$. The action of H_X on ψ will then be defined by mapping each summand (t, H_t, ξ_t) in $H_X(\Gamma)$ to the corresponding $(\widetilde{\psi}(t), H_{\widetilde{\psi}(t)}, \xi_{\widetilde{\psi}(t)})$ in $H_X(\Delta)$. The association is clearly functorial, and $U_X H_X = F U_X$, therefore we can choose α'_X to be the identity.

Example 4.20 The intuition behind the definition of H_X is very simple. If, for example, $t_1, t_2 \in G(X)$ are two term graphs obtained as the image of g_1 and g_2 in the coalgebra



then the term $\mathtt{A}(t_1, t_2)$ in $F(G(X))$ is obtained as the image of the root of the following coalgebra, where together with the variables we have included also the whole coalgebras G_i.



Given a map $f\colon X \longrightarrow GY$ factoring as above, it is an easy but lengthy computation that the two functors $\Lambda(f)H_X$ and $H_Y\Lambda(f)$ are equal, as well as the natural transformations $f^\Lambda \Diamond \alpha'_X$ and $\alpha'_Y \Diamond F(f^\Lambda)$. Therefore, we can consider the identity 2-cell in between them, and by Corollary 3.21 we know that G is F-guarded.

4.3 Some Considerations

In this chapter we gave two examples of applications of the results presented in Chapter 3. In our opinion, these are sufficient to motivate the introduction of the machinery. Although the notation makes instantiation of the theorem quite laborious, the concepts behind the construction are relatively easy, and it is not difficult to understand what choice of maps is the most appropriate. We hope that the examples provided along the computation support this comment, having showed to the reader what intuition lead us through the work.

Although the monad of term graphs is intrinsically related to the category of sets (but it is not excluded that one can extend it to some concrete category, like Pre, maybe recovering the notion of term graph rewriting), the construction of the rational monad works in any lfp category, also for functors which do not arise from a signature. The wide variety of applications have not been investigated for the purpose of this thesis, but they should cover many areas of mathematics and computer science, where some finitary operation is iterated to get some result in the limit. Only one example is exploited in the next Chapter, where we shall reinterpret some classical notion in the theory of term rewriting systems by means of the rational monad and the monad of infinite terms.

Chapter 5

Parallel Rewriting

In this chapter we present a nice and simple application to term rewriting of the theory we have developed so far. In detail, we shall recall the monadic semantic for term rewriting as proposed by Christoph Lüth and Neil Ghani in [42, 43]. In this framework, we have a natural way of identifying those rewrites which can be performed in parallel. When working with finite terms, considering parallel rewrites does not enrich the multiple-step rewrite relation; however, things change when one considers infinite terms.

Suppose we have the rule $\{x\} \vdash \rho \colon \mathtt{A}(x) \to \mathtt{B}(x)$, and consider the infinite term $\mathtt{A}^\omega = \mathtt{A}(\mathtt{A}(\mathtt{A}(\ldots)))$. Then, intuitively, it makes sense to consider the simultaneous replacement of all the A's with B's as a single rewrite step, leading from $\mathtt{A}^\omega$ to $\mathtt{B}^\omega$. Clearly, such a rewrite is not derivable in finitely many steps from the given rule. What we need is a coinductive interpretation of ρ, and that is what we gain by considering the monad of infinite terms, which is pointwise the final coalgebra over the functor building terms over our signature. In other words, we shall replace the free monad T in the classical monadic semantics of rewriting by the free completely iterative monad T^{ν} on the same endofunctor.

In [18] Corradini and Gadducci use infinite parallel rewriting to define rational rewriting. In our setting, it is perfectly reasonable to replace T^{ν} by the rational monad R in the relevant places, and we can easily provide a monadic semantic for all these different kinds of rewriting.

This chapter is organised as follows. In the first section we shall recall the presentation of a term rewriting system in the Kelly-Power style, exploring in detail how the rewrites are generated and how parallel rewrites are modelled. This will lay down the intuition needed for the second section, where the theory is extended in ordered to capture infinite parallel rewriting. In the third section, rational rewriting is tackled. In each case, we provide a categorical semantic, and we show that it agrees with the standard notions as introduced in [18].

5.1 Categorical Term Rewriting

In Example 1.45 on page 58, we gave an example of how to model a TRS categorically, by means of a monad modelling an equational theory on Pre. The same technique can be applied in order to model any term rewriting system. We shall now explain how to get from a TRS $\langle \Sigma, \mathcal{R} \rangle$ to an equational theory over Pre, presented by means of a finitary monad on that category. In other words, we will describe a monad $\mathsf{T}_{\langle \Sigma, \mathcal{R} \rangle}$ on Pre such that $T_{\langle \Sigma, \mathcal{R} \rangle} X$ is the preorder defined by the rewrite relations induced by the rules in $\mathcal{R}$ over the set of terms $T_{\Sigma} X$. In fact, the presentation allows for the variables in

X to form a preorder, but the usual notion can be recovered by taking the preorder to be discrete. The content of this section is presented in Christoph Lüth's PhD thesis, to which we refer for further reading [42].

So, let Σ be a finitary **Set**-signature, and $\mathcal{R} = \{X_i \vdash \rho_i : t_i \to s_i\}_{i \in I}$ a set of rules. We define a new finitary signature Σ' on **Pre**. This will be given by presenting for each arity, a preorder of function symbols. Finitary arities, in **Pre** (as a category enriched over itself), are given by preorders with a finite underlying set. In particular, Σ' will be defined only on those finite preorders which are discrete (we shall denote them by n, where n is the number of elements), meaning by this that $\Sigma'(p)$ is the empty preorder for any other arity p. On the arity n, we shall define

$$\Sigma'(n) = \Sigma(n) + \sum_{i \in I, |X_i| = n} (l_i \to r_i). \tag{5.1}$$

In other words, there is a term constructor for each n-ary term constructor in Σ, and no rewrite is introduced amongst them. Further, we introduce two more function symbols for each rule in $\mathcal{R}$. The symbols l_i and r_i are intended to stand for the left and right handside of the rewrite rule ρ_i, respectively, and for that reason they are related by the preorder. These last symbols are the ones which add some rewrite when building terms, as the only rewrites in our signature are defined amongst them. We shall often refer to the collection of the l_i and r_i symbols as *place-holders*.

Equations are our means of forcing the equality between the place-holders and the terms which they stand for. For each rule ρ_i in $\mathcal{R}$, we need an equation to set $l_i = t_i$ and another one to set $r_i = s_i$. In the framework of Kelly and Power [37], equations are described by another signature, and their left and right handsides are specified by two monad morphisms λ and ρ. Our signature for equations will be denoted by E, and is again defined as non-empty only on discrete finite preorders. Moreover, it will define no ordering between the term constructors. The actual definition is

$$E(n) = \sum_{i \in I, |X_i| = n} (e_i^l \quad e_i^r). \tag{5.2}$$

The symbol e_i^l stands for the equation $l_i = t_i$, whereas e_i^r stands for $r_i = s_i$ $(i \in I)$. Given these signatures, we can form the two free monads on them, which we shall denote by $\mathsf{T}_{\Sigma'}$ and T_E. Since T_E is free over E, the aforementioned monad morphisms $\lambda, \rho \colon \mathsf{T}_E \longrightarrow \mathsf{T}_{\Sigma'}$ are determined by their restriction $\lambda', \rho' \colon E \longrightarrow T_{\Sigma'}JI$, which we can easily define by putting, for each $i \in I$,

$$\begin{aligned} \lambda'(e_i^l) &= l_i & \qquad \lambda'(e_i^r) &= r_i \\ \rho'(e_i^l) &= t_i & \qquad \rho'(e_i^r) &= s_i. \end{aligned} \tag{5.3}$$

Having defined this parallel pair of monad morphism, we can now consider their coequaliser $\mathsf{T}_{\langle\Sigma,\mathcal{R}\rangle}$:

$$\begin{tikzcd} \mathsf{T}_E \arrow[r, shift left, "\lambda"] \arrow[r, shift right, "\rho"'] & \mathsf{T}_{\Sigma'} \arrow[r, "q"] & \mathsf{T}_{\langle\Sigma,\mathcal{R}\rangle}. \end{tikzcd} \tag{5.4}$$

We know by [34, Section 26], that the preorder $T_{\langle\Sigma,\mathcal{R}\rangle}X$ is the free $\mathsf{T}_{\Sigma'}$-algebra over X making λ and ρ equal (i.e. the free object M on X having an algebra structure (M, α) for the monad $\mathsf{T}_{\Sigma'}$ and such that $\alpha\lambda_M = \alpha\rho_M$). Lüth, in his PhD thesis [42], gave a concrete description of the preorder $T_{\langle\Sigma,\mathcal{R}\rangle}X$. While he was making use of the Kelly-Power framework, he did not make it explicit, and in particular he did not show how that preorder satisfies the required universal property. Since we shall need that property also in the next sections, we will investigate it thoroughly, and for this we find convenient to give a different description of the preorder.

We know that $T_{\Sigma'}X$ is the free preorder generated by Σ' on X. Its carrier is the set of finite terms built over Σ', and a term t rewrites to a term s if and only if t and s differ at most for some occurrences of l_i-symbols in t which are replaced by the matching r_i's in s. So, for example, the term $\mathtt{A}(\mathtt{B}(l_1(x), l_2))$ rewrites to$\mathtt{A}(\mathtt{B}(r_1(x), r_2))$.

Proposition 5.1 *The preorder $T_{\langle\Sigma,\mathcal{R}\rangle}X$ is a quotient of $T_{\Sigma'}X$ under the relation $\sim$ inductively defined by the following clauses:*

$$\text{[REFL]}\ \frac{t \in T_{\Sigma'}X}{t \sim t} \qquad \text{[CONG]}\ \frac{t_1 \sim s_1, \ldots, t_n \sim s_n, f \in \Sigma'(n)}{f[t_1/x_1, \ldots, t_n/x_n] \sim f[s_1/x_1, \ldots, s_n/x_n]}$$

$$\text{[SYM]}\ \frac{s \sim t}{t \sim s} \qquad \text{[SUBL]}\ \frac{\sigma: X_i \longrightarrow T_{\Sigma'}X}{\sigma(l_i) \sim \sigma(t_i)}\ i \in I$$

$$\text{[TRANS]}\ \frac{t \sim s,\ s \sim u}{t \sim u} \qquad \text{[SUBR]}\ \frac{\sigma: X_i \longrightarrow T_{\Sigma'}X}{\sigma(r_i) \sim \sigma(s_i)}\ i \in I$$

Proof. Let $(M, \to)$ be the defined quotient $q: T_{\Sigma'}X \longrightarrow M$. We are going to prove its universal property. It is clear by [CONG] that all function symbols preserve the relation $\sim$; therefore, an algebra structure α for the endofunctor $T_{\Sigma'}$ can be induced on M. The fact that all functions are monotone is trivial, since the order on M is the one induced by q. Laws [SUBL] and [SUBR] ensure that (M, α) is indeed an algebra for the monad $\mathsf{T}_{\Sigma'}$ and that the equality $\alpha\lambda_M = \alpha\rho_M$ holds. The unit $\eta': X \longrightarrow M$ is defined by the composite

$$X \xrightarrow{\eta_X} T_{\Sigma'}X \xrightarrow{q} M.$$

We now need to show that the 4-tuple $(\eta', M, \rightarrow, \alpha)$ is universal, i.e. that given any other $\mathsf{T}_{\Sigma'}$-algebra $(M', \geq, \beta)$ such that $\beta\lambda_{M'} = \beta\rho_{M'}$, and any map $\phi\colon X \longrightarrow M'$ in Pre, there is a unique $T_{\Sigma'}$-algebra morphism ϕ' from M to M' such that $\phi'\eta' = \phi$.

Because (M', β) is a $\mathsf{T}_{\Sigma'}$-algebra an $(T_{\Sigma'}X, \mu)$ is the free such, ϕ induces a unique morphism $\widetilde{\phi}$:

$$\begin{tikzcd}
T^2_{\Sigma'}X \arrow[r, "T_{\Sigma'}\widetilde{\phi}"] \arrow[d, "\mu"'] & T_{\Sigma'}M' \arrow[d, "\beta"] \\
T_{\Sigma'}X \arrow[r, "\widetilde{\phi}"'] & M'.
\end{tikzcd} \tag{5.5}$$

We complete the proof if we show that $\widetilde{\phi}$ factors through M, i.e. that $\widetilde{\phi}(t) = \widetilde{\phi}(s)$ whenever $t \sim s$ in $T_{\Sigma'}X$. We prove it by induction on the clauses defining $\sim$. For [REFL], [SYM] and [TRANS] it is obvious. The fact that, for a substitution $\sigma\colon X_i \longrightarrow T_{\Sigma'}X$, $\widetilde{\phi}(\sigma(l_i)) = \widetilde{\phi}(\sigma(t_i))$ and $\widetilde{\phi}(\sigma(r_i)) = \widetilde{\phi}(\sigma(s_i))$ follows from the fact that β satisfies the equations. In fact, $\sigma(l_i)$ is the result of applying μ to the term $l_i(\sigma_1, \ldots, \sigma_n)$, where σ_i is the term $\sigma(x_i)$. By (5.5) we then have

$$\begin{aligned}
\widetilde{\phi}(\mu(l_i(\sigma_1, \ldots, \sigma_n))) &= \beta(l_i(\widetilde{\phi}(\sigma_1), \ldots, \widetilde{\phi}(\sigma_n))) \\
&= \beta(t_i[\widetilde{\phi}(\sigma_1)/x_1, \ldots, \widetilde{\phi}(\sigma_n)/x_n]) \\
&= \widetilde{\phi}(\mu(t_i[\sigma_1/x_1, \ldots, \sigma_n/x_n]))
\end{aligned}$$

and similarly for r_i and s_i $(i \in I)$. Finally, if $\widetilde{\phi}(t_1) = \widetilde{\phi}(s_1), \ldots, \widetilde{\phi}(t_n) = \widetilde{\phi}(s_n)$ and f is an n-ary function symbol from Σ', then we want to show that

$\widetilde{\phi}(f(t_1,\dots,t_n)) = \widetilde{\phi}(f(s_1,\dots,s_n))$. This follows again from (5.5), since

$$\begin{aligned}\widetilde{\phi}(f(t_1,\dots,t_n)) &= \widetilde{\phi}(\mu(f(t_1,\dots,t_n))) \\ &= \beta(f(\widetilde{\phi}(t_1),\dots,\widetilde{\phi}(t_n))) \\ &= \beta(f(\widetilde{\phi}(s_1),\dots,\widetilde{\phi}(s_n))) \\ &= \widetilde{\phi}(f(s_1,\dots,s_n)),\end{aligned}$$

where we write $f(t_1,\dots,t_n)$ to indicate both the term in $T^2_{\Sigma'}X$ and its image under μ. □

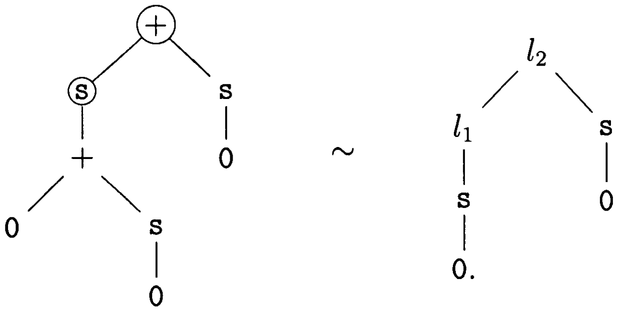
Given the way we defined the rules, it is quite easy to show that in any $\sim$-equivalence class in $T_{\Sigma'}X$ there is precisely one term from $T_\Sigma X$ (i.e. a term built solely on Σ-symbols, with no place-holders). The quotient $T_{\langle\Sigma,\mathcal{R}\rangle}X$ can therefore be seen as a preorder on $T_\Sigma X$, and it is not hard to see that it is the one generated by the rules. A term t in $T_\Sigma X$ rewrites to $s \in T_\Sigma X$ if and only if $t \sim t'$ and $s \sim s'$ in $T_{\Sigma'}X$, where t' is a finite term with some occurrences of some l_i's and s' is that same term with those occurrences replaced by the corresponding r_i's. It is then clear that $t' \dashrightarrow s'$ in $T_{\Sigma'}X$, and the rewrite gets inherited by their equivalence classes in the quotient, hence determining $t \dashrightarrow s$.

The process of choosing an equivalent term t' for a Σ-term t, so that in t' some place-holders l_i appear, can be seen as *highlighting* some disjoint redexes in t. The act of replacing the l_i's with the matching r_i's is nothing else but performing a *parallel reduction* of all the highlighted redexes.

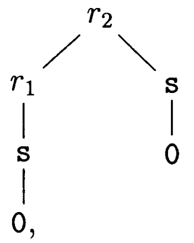
We can recover the single step rewriting relation by considering those reductions which arise by highlighting precisely one occurrence of some l_i in the term t. This way we prove the following:

Proposition 5.2 *The preorder $T_{\langle\Sigma,\mathcal{R}\rangle}X$ is the reduction relation defined by $\mathcal{R}$ on the set of Σ-terms with variables in X.*

Example 5.3 Let $\langle\Sigma,\mathcal{R}\rangle$ be the TRS in Example 1.45. We described already in there the signatures Σ' and E. Let's consider the term $\mathtt{s(0+s(0))+s(0)}$ in $T_\Sigma X$. We can highlight two redexes (whose roots we circle in a tree-like representation of the term) by choosing the equivalent Σ'-term



In $T_{\Sigma'}X$, this equivalent term rewrites to



and by replacing the r_i's with the corresponding terms we get the following rewrite in $T_\Sigma X$

$$\mathtt{s(0+s(0))+s(0)} \to \mathtt{s(s(0)+s(0))}.$$

We can think of this as a single step reduction of the parallel rewrite determined by the two redexes highlighted above.

5.2 Infinite Parallel Rewriting

In the finite case, parallel rewrites are not particularly important, in a TRS. Noting that a particular set of redexes is disjoint means that we can perform then in *parallel*, and consider that as a single-step rewrite. However, this adds no more expressive power to the TRS than it already had. We are just putting a hat on some of the rewrites. The situation changes when one considers infinite terms.

Given a signature Σ and a set of rules $\mathcal{R}$, these clearly induce a rewriting system on the set $T^{\nu}_{\Sigma}X$ of finite and infinite Σ-terms over X. This will be the reflexive and transitive closure of the single-step rewrite relation defined analogously to the finite case (1.4).

Because the rewrite relation is determined by a transitive closure, though, it preserves a finitary character, despite being performed on infinite terms. So, for example, if we have a rule

$$\{x\} \vdash \rho \colon \mathtt{A}(x) \to \mathtt{B}(x), \tag{5.6}$$

we can show that

$$\mathtt{A}^{\omega} \longrightarrow \mathtt{B}^{n}\mathtt{A}^{\omega} \quad \text{for any} \quad n \in \mathbb{N}$$

but we can not show that $\mathtt{A}^{\omega} \to \mathtt{B}^{\omega}$. Still, this would be desirable, and not just because it is a clearly harmless extension of the calculus to capture infinite rewrites of disjoint redexes. We can in fact build the chain of reductions

$$\mathtt{A}^{\omega} \longrightarrow \mathtt{B}\,\mathtt{A}^{\omega} \longrightarrow \mathtt{B}^{2}\mathtt{A}^{\omega} \longrightarrow \mathtt{B}^{3}\mathtt{A}^{\omega} \longrightarrow \mathtt{B}^{4}\mathtt{A}^{\omega} \longrightarrow \dots$$

so, for some continuity argument, it would be reasonable to assume that $\mathtt{A}^{\omega}$ rewrites to the limit of the sequence too, and that is precisely $\mathtt{B}^{\omega}$. Last, but not least, if we think of the rewrite relation as a computation, then the sequence above shows an infinite computation of $\mathtt{A}^{\omega}$ which, after n steps, has fixed the first n elements of the result to be $\mathtt{B}^{n}$. It therefore makes sense to say that the result of the whole computation should be the limit of the sequence.

The three arguments we have briefly sketched here do in fact reflect three different approaches to the problem of extending rewriting to infinite terms. The most general way of extending it is to say that in presence of a sequence of single-step rewrites

$$t_0 \longrightarrow t_1 \longrightarrow t_2 \longrightarrow t_3 \longrightarrow \ldots$$

where $(t_i)_{i \in \mathbb{N}}$ is a Cauchy sequence of terms with a limit t, each t_i rewrites to t, despite of the properties of the reduction sequence. This extension, though, is easily seen to break confluence; for this reason people introduced the notion of *strongly converging reduction sequence*, where not only one requires the t_i's to form a Cauchy sequence, but also that the rewrites take place at increasing depth along the sequence [38].

Strongly convergent infinite rewriting matches the third argument above. The first one, instead, gave rise to what is called *infinite parallel rewriting* . There, we start with an infinite term t and a set Φ of redexes in an orthogonal TRS, and we consider a chain $(t_n)_{n \in \mathbb{N}}$ of finite approximations of t (in other words each t_i is a truncation of the term t_{i+1}) such that for any n each redex in Φ is either completely inside the term t_n or does not appear in it at all (i.e. t_n is a truncation of t at a node which occurs above the redex). We write Φ_i for the set of redexes from Φ which appear in t_i, and we call s_i the term obtained by performing the (finite) parallel reduction of t_i via Φ_i. It can be shown that the terms $(s_n)_{n \in \mathbb{N}}$ form a chain, and we put its least upper bound s to be the result of the infinite parallel reduction of t via Φ (for a precise definition of the relation, see [18]). The idea is that s is the result of simultaneously reducing all the redexes in Φ.

In this section we shall move from the presentation of a TRS as a monad over Pre in order to catch the notion of infinite parallel rewriting amongst infinite terms.

Given a signature Σ and a set of rules $\mathcal{R}$, we define the signature Σ' on Pre as before (5.1). Instead of considering the free monad over it, though, we consider the monad $\mathsf{T}^{\nu}_{\Sigma'}$ as defined in Section 2.4. Its action on a preorder X gives the preorder coinductively defined by Σ'. The carrier is the set of finite and infinite trees built over the Σ-symbols and the place-holders with variables from x. Whenever in a term t there is some occurrence of some l_i's, whether finitely or infinitely many, t rewrites to the term s, where those same occurrences are replaced by the corresponding r_i's. Notice how, by working in Pre, we get at once the extension of the terms *and* the rewrites to the infinite case, rather than defining the finite reduction preorder first and then having to extend it.

Remark 5.4 In fact, we are working with monads in the enriched setting. For this reason, we should check in various places that our functors preserve the induced preorder relation on the hom-sets. We shall omit such checks in the presentation, as they are completely straightforward and not insightful.

As before, we are interested in replacing the place-holders by Σ-terms which they are meant to represent, in order to get a rewriting system on the set of infinite Σ-terms. This can be achieved again by imposing equations, and the signature we are going to use for denoting them is the same E we defined in (5.2). However, several different parallel pairs can be built, using E, and we have to take care of choosing the correct one. First of all, let's recall that there is a unique morphism of $F_{\Sigma'}$-guarded monads $\iota_{\Sigma'}\colon \mathsf{T}_{\Sigma'} \longrightarrow \mathsf{T}^{\nu}_{\Sigma'}$. Given the signature E, we can therefore consider the parallel pair

$$\begin{tikzcd}
\mathsf{T}_E \arrow[r, shift left, "\lambda"] \arrow[r, shift right, "\rho"'] & \mathsf{T}_{\Sigma'} \arrow[r, "\iota_{\Sigma'}"] & \mathsf{T}^{\nu}_{\Sigma'}.
\end{tikzcd}$$

However, the coequaliser of this pair would not work for our need, for it would only be able to identify finitely many place-holders with their matching terms. In order to identify at once infinitely many l_i's (respectively r_i's) with the corresponding t_i's (resp. s_i's), we need to consider an extension of T_E which computes infinite terms. The obvious coequaliser then becomes the one on the bottom row of the following diagram:

$$\begin{tikzcd}
\mathsf{T}_E \arrow[r, shift left, "\lambda"] \arrow[r, shift right, "\rho"'] \arrow[d, "\iota_E"'] & \mathsf{T}_{\Sigma'} \arrow[d, "\iota_{\Sigma'}"] & \\
\mathsf{T}^{\nu}_E \arrow[r, shift left, "\lambda^{\nu}"] \arrow[r, shift right, "\rho^{\nu}"'] & \mathsf{T}^{\nu}_{\Sigma'} \arrow[r, "q"'] & \mathsf{T}.
\end{tikzcd} \tag{5.7}$$

Here, in order to extend λ and ρ to monad morphisms λ^{ν} and ρ^{ν} from T^{ν}_E to $\mathsf{T}^{\nu}_{\Sigma'}$ we use the universal property of T^{ν}_E, i.e. that of being the free completely iterative monad on F_E. It is not difficult to show that each of the composites $\iota_{\Sigma'}\lambda'$ and $\iota_{\Sigma'}\rho'$ from F_E to $\mathsf{T}^{\nu}_{\Sigma'}$ is an ideal natural transformation (in the sense of Adámek et al. [1]) precisely when both handsides of the equations are guarded terms (i.e. not just variables), and in that case, they determine the depicted morphisms. We shall discuss later what this condition means on the level of the term rewriting system.

Unfortunately, this is still not what we want. The coequaliser in (5.7) can only identify an infinite sequence of place-holders with their matching terms if they are not interleaved with any term constructor from Σ. For example, in presence of a rule such as (5.6) above, we would not be able to prove that $(l\,\mathtt{B})^\omega$ is equivalent to $(\mathtt{A}\,\mathtt{B})^\omega$ (where l is the place-holder for the left handside of ρ). In order to overcome this hindrance, we have to look back at the finite case a bit more carefully.

First of all, let's recall that, as a special case of [53, Theorem 1], we have the following result.

Lemma 5.5 *If* $A \underset{g}{\overset{f}{\rightrightarrows}} B \xrightarrow{q} D$ *is a coequaliser in a category* C*, then, for any* C *and any* $h\colon C \longrightarrow B$ *in* C*,* q *is the coequaliser of the parallel pair*

$$A + C \underset{[g,h]}{\overset{[f,h]}{\rightrightarrows}} B.$$

Now, because the free monad construction is a left adjoint, we have that $\mathsf{T}_E \oplus \mathsf{T}_{\Sigma'} = \mathsf{T}_{E+\Sigma'}$, where $\oplus$ is the coproduct in the category of accessible monads, and taking $h = \mathsf{id}_{\mathsf{T}_{\Sigma'}}$ in the above lemma we get that the coequaliser in (5.4) is the same as the following:

$$\begin{tikzcd} \mathsf{T}_{E+\Sigma'} \arrow[r, shift left, "{[\lambda,\mathsf{id}]}"] \arrow[r, shift right, "{[\rho,\mathsf{id}]}"'] & \mathsf{T}_{\Sigma'} \arrow[r, "q"] & \mathsf{T}_{\langle\Sigma',\mathcal{R}\rangle}. \end{tikzcd}$$

A similar argument is no longer valid when considering $\mathsf{T}^\nu_{E+\Sigma'}$, since this monad is not the same as $\mathsf{T}^\nu_E \oplus \mathsf{T}^\nu_{\Sigma'}$. Instead of considering the coequaliser in (5.7), we can then consider the one below:

$$\begin{tikzcd} \mathsf{T}^\nu_{E+\Sigma'} \arrow[r, shift left, "\lambda^\nu"] \arrow[r, shift right, "\rho^\nu"'] & \mathsf{T}^\nu_{\Sigma'} \arrow[r, "q"] & \mathsf{T}^{//}_{\langle\Sigma',\mathcal{R}\rangle}, \end{tikzcd}$$

where the monad morphisms λ^ν and ρ^ν are defined as follows.

The monad $\mathsf{T}^\nu_{E+\Sigma'}$ is the free completely iterative one on the finitary endofunctor $F_{E+\Sigma'}$, therefore by [1, Theorem 4.14] we get the monad morphisms λ^ν and ρ^ν by giving two guarded (ideal, in their notations) natural transformations λ' and ρ' from $F_{E+\Sigma'}$ to $\mathsf{T}^\nu_{\Sigma'}$ factoring through $F_{\Sigma'}\mathsf{T}^\nu_{\Sigma'}$. These are clearly determined once we describe the action of λ' and ρ' on each term constructor in $E + \Sigma'$. The action on a Σ'-symbol is in both cases the identity, whereas on the place-holders we define them as in (5.3).

Because the left handsides of all the equations are the place-holders l_i and r_i $(i \in I)$, the morphism λ' is always guarded, and we can safely extend it to λ^ν. As for ρ', we have to require that the right handsides of the equations, i.e. the t_i's and s_i's of the rules in $\mathcal{R}$, are guarded terms. For this reason, we restrict our attention to TRS's where both sides of the rules are not variables. In particular, we are avoiding collapsing rules. It is not surprising that the categorical constraints on developing our machinery match some well-known trouble-making notion in the theory of rewriting. In fact, if we were to accept a collapsing rule such as $\{x\} \vdash \rho\colon \mathtt{A}(x) \to x$, then, clearly, for each $n \in \mathbb{N}$ we would have $\mathtt{A}^n(x) \to x$. However, the term $\mathtt{A}^\omega$ would have no infinite parallel reduction to anything, because after having removed all the occurrences of $\mathtt{A}$ we do not know what to output. This pathological example is paradigmatic, and a term like $\mathtt{A}^\omega$ above is often called a *collapsing tower* . Rewriters usually like to consider the set of infinite terms as a complete partial order, with a bottom element $\perp$, which they think of as a symbol representing a *meaningless* term. They would then say that $\mathtt{A}^\omega \to \perp$, with the underlying intuition that $\mathtt{A}^\omega$ has a meaningless computation. In our context, we do not have a bottom element, so our solution is to ban such rewrites, so that meaningless computations simply do not exist.

Remark 5.6 Note that λ' always factors as

$$\begin{tikzcd} F_{E+\Sigma'} \arrow[r, "\lambda'"] \arrow[d, "\phi"'] & \mathsf{T}^{\nu}_{\Sigma'} \\ \mathsf{T}_{E+\Sigma'} \arrow[r, "\lambda"'] & \mathsf{T}_{\Sigma'} \arrow[u, "\iota_{\Sigma'}"'] \end{tikzcd}$$

where ϕ is the universal arrow from $F_{E+\Sigma'}$ to the free monad over it and λ is the natural transformation defined in (5.4).

If the terms appearing in the rewrite rules are all finite, then we can factorise ρ' in the same way, and the parallel pair λ^{ν}, ρ^{ν} is the extension to $\mathsf{T}^{\nu}_{E+\Sigma'}$ of the pair $\iota_{\Sigma'}\lambda$, $\iota_{\Sigma'}\rho$. However, there is no formal reason to force such a factorisation for ρ, and we can allow ρ' to reach some truly infinite terms, so long as they are all guarded. In other words, once we consider infinite terms, there is no reason to maintain the rules finite; in fact, we can allow them to rewrite infinite terms as well.

So, with the assumption that both sides of each rule in $\mathcal{R}$ are guarded terms, we can define the parallel pair

$$\begin{tikzcd} \mathsf{T}^{\nu}_{E+\Sigma'} \arrow[r, shift left, "\lambda^{\nu}"] \arrow[r, shift right, "\rho^{\nu}"'] & \mathsf{T}^{\nu}_{\Sigma'} \end{tikzcd}$$

and consider its coequaliser, which we shall indicate by $\mathsf{T}^{//}_{\langle\Sigma,\mathcal{R}\rangle}$. We now want to show that, for an orthogonal TRS $\langle\Sigma,\mathcal{R}\rangle$ and a discrete preorder X of variables, $T^{//}_{\langle\Sigma,\mathcal{R}\rangle}X$ is the infinite parallel rewriting preorder induced by $\mathcal{R}$ on the set of finite and infinite Σ-terms over X.

Remark 5.7 As we already stressed in Remark 5.6, we are gaining here some generality with respect to the formulation proposed in [18].

We already discussed the fact that the rules can define reductions of infinite terms. Moreover, this categorical formulation allows our TRS's to inherit some reductions from a pre-existing preorder on the set of variables which the terms are built upon.

When describing the equivalence of the two approaches, we shall restrict ourselves to the standard assumptions, but in general we are providing an extension of the classical theory.

In order to understand what $T^{//}_{\langle\Sigma,\mathcal{R}\rangle}X$ looks like, we have to refer again to its universal property: it is the free $\mathsf{T}^{\nu}_{\Sigma'}$-algebra on X satisfying the equations. We are now going to give a concrete description of it. In fact, this might look a bit surprising to a reader familiar with category theory, and in in particular with coequalisers of monads, which are usually far from being computed pointwise, but in this case this is precisely what happens. Let's consider the following coequaliser in Pre:

$$\begin{tikzcd}
T^{\nu}_{E+\Sigma'}X \arrow[r, shift left, "\lambda^{\nu}_X"] \arrow[r, shift right, "\rho^{\nu}_X"'] & T^{\nu}_{\Sigma'}X \arrow[r, "q"] & S.
\end{tikzcd} \tag{5.8}$$

We want to define a $\mathsf{T}^{\nu}_{\Sigma'}$-algebra structure on S and show that it satisfies the equations. Moreover, we shall show that it is the free such object on X, thus proving that S is the object $T^{//}_{\langle\Sigma',\mathcal{R}\rangle}X$.

First of all, note that, by the way coequalisers are computed in Pre, S has as carrier the set $T^{\nu}_{\Sigma'}X/\sim$, where $\sim$ is the equivalence relation generated by the pairs of terms in the image of $T^{\nu}_{E+\Sigma'}X$ along the pairing of functions

$\langle \lambda^\nu_X, \rho^\nu_X \rangle$. For such a pair (t, s), the term t has a (possibly infinite) set of occurrences of place-holder symbols l_i and r_i, and s has those same occurrences replaced by the matching t_i's and s_i's, respectively. The preorder on S is the smallest on $T^\nu_{\Sigma'}X/\sim$ making q monotone.

Note also that the endofunctor $T^\nu_{\Sigma'}$ on Pre induces one on Set which we shall denote by the same symbol. This will map a set X to the underlying set of the image of X (considered as a discrete preorder) through $T^\nu_{\Sigma'}$.

We now define three maps in Set which will play an important role in our argument. The map $f\colon S \longrightarrow T^\nu_{\Sigma'}X$ takes the equivalence class $\bar{t}$ in S of a term t in $T^\nu_{\Sigma'}X$ to the unique representative of t with no occurrence of l_i or r_i symbols. The function $g\colon T^\nu_{\Sigma'}X \longrightarrow T^\nu_{E+\Sigma'}X$ takes a term t to the term obtained by replacing each occurrence of an l_i (respectively r_i) with the corresponding equation symbol e^l_i (resp. e^r_i), leaving all Σ-symbols unchanged. Finally, the map $h_X\colon T^\nu_{\Sigma'}X \longrightarrow T^\nu_{E+\Sigma'}X$ is the X-th component of the monad morphism $h\colon \mathsf{T}^\nu_{\Sigma'} \longrightarrow \mathsf{T}^\nu_{E+\Sigma'}$ determined by the natural transformation $\mathsf{inr}\colon \Sigma' \longrightarrow E + \Sigma'$. This will map a Σ'-term t to itself, considered as a term over the signature $E + \Sigma'$, and clearly makes λ^ν and ρ^ν equal:

$$\lambda^\nu_X h = \mathsf{id} = \rho^\nu_X h. \tag{5.9}$$

Furthermore, f and g make (5.8) a contractible coequaliser (the dual notion of that in Definition 2.21), since $\lambda^\nu_X g = \mathsf{id}$, $\rho^\nu_X g = fq$, $q\lambda^\nu_X = q\rho^\nu_X$ and $qf = \mathsf{id}$:

$$\begin{tikzcd} T^\nu_{E+\Sigma'}X \arrow[r, shift left=3, "\lambda^\nu_X"] \arrow[r, shift right=3, "\rho^\nu_X"'] & T^\nu_{\Sigma'}X \arrow[l, "g" description] \arrow[r, shift left, "q"] & S. \arrow[l, shift left, "f"] \end{tikzcd} \tag{5.10}$$

Because absolute coequalisers are preserved by all functors, we get, by applying $T^\nu_{\Sigma'}$, that

$$\begin{tikzcd}
T^\nu_{\Sigma'}T^\nu_{E+\Sigma'}X \arrow[r, shift left, "T^\nu_{\Sigma'}\lambda^\nu_X"] \arrow[r, shift right, "T^\nu_{\Sigma'}\rho^\nu_X"'] & {T^\nu_{\Sigma'}}^2X \arrow[r, "T^\nu_{\Sigma'}q"] & T^\nu_{\Sigma'}S
\end{tikzcd} \tag{5.11}$$

is a coequaliser. We can put on $T^\nu_{\Sigma'}S$ the smallest preorder relation making $T^\nu_{\Sigma'}q$ monotone (that is, in fact, the same preorder as the one determined by the Pre-endofunctor $T^\nu_{\Sigma'}$ on S), and we get that $T^\nu_{\Sigma'}S$ is a coequaliser in Pre.

We now join (5.11) and (5.8) in order to get a $\mathsf{T}^\nu_{\Sigma'}$-algebra structure α on S.

$$\begin{tikzcd}
T^\nu_{\Sigma'}T^\nu_{E+\Sigma'}X \arrow[r, shift left, "T^\nu_{\Sigma'}\lambda^\nu_X"] \arrow[r, shift right, "T^\nu_{\Sigma'}\rho^\nu_X"'] \arrow[d, "h_{T^\nu_{E+\Sigma'}X}"'] & {T^\nu_{\Sigma'}}^2X \arrow[r, "T^\nu_{\Sigma'}q"] \arrow[dd, "\mu_X"] & T^\nu_{\Sigma'}S \arrow[dd, dashed, "\alpha"] \\
{T^\nu_{E+\Sigma'}}^2X \arrow[d, "\overline{\mu}_X"'] & & \\
T^\nu_{E+\Sigma'}X \arrow[r, shift left, "\lambda^\nu_X"] \arrow[r, shift right, "\rho^\nu_X"'] & T^\nu_{\Sigma'}X \arrow[r, "q"] & S.
\end{tikzcd}$$

In the Pre-diagram above, since λ^ν and ρ^ν are monad morphisms, we have

$$\begin{aligned}
q\mu_X T^\nu_{\Sigma'}\lambda^\nu_X &= q\mu_X(\lambda^\nu)^2_X h_{T^\nu_{E+\Sigma'}X} \\
&= q\lambda^\nu_X\overline{\mu}_X h_{T^\nu_{E+\Sigma'}X} \\
&= q\rho^\nu_X\overline{\mu}_X h_{T^\nu_{E+\Sigma'}X} \\
&= q\mu_X T^\nu_{\Sigma'}\rho^\nu_X
\end{aligned}$$

where $\overline{\mu}$ is the multiplication of the monad $\mathsf{T}^\nu_{E+\Sigma'}$ and the first equality holds because of (5.9). Therefore, because the top row of the diagram is a coequaliser, there is a unique map $\alpha\colon T^\nu_{\Sigma'}S \longrightarrow S$ such that $\alpha T^\nu_{\Sigma'}q = q\mu_X$. Notice that, once we have proved that α is an algebra for the monad $\mathsf{T}^\nu_{\Sigma'}$, we have for free from this last equality that q is an algebra homomorphism.

To show that α is an algebra for $\mathsf{T}^{\nu}_{\Sigma'}$, we have to show that $\alpha\eta_S = \mathsf{id}_S$ and $\alpha T^{\nu}_{\Sigma'}\alpha = \alpha\mu_S$. For the unit identity, it is easy to see that

$$\alpha\eta_S q = \alpha T^{\nu}_{\Sigma'}(q)\eta_{T^{\nu}_{\Sigma'}X} = q\mu_X\eta_{T^{\nu}_{\Sigma'}X} = q,$$

hence $\alpha\eta_S = \mathsf{id}_S$ because q is epic.

As for the multiplication identity, we can form the following diagram

$$\begin{tikzcd}
T^{\nu}_{\Sigma'}{}^2 T^{\nu}_{E+\Sigma'}X \arrow[r, shift left, "T^{\nu}_{\Sigma'}{}^2\lambda^{\nu}_X"] \arrow[r, shift right, "T^{\nu}_{\Sigma'}{}^2\rho^{\nu}_X"'] & T^{\nu}_{\Sigma'}{}^3X \arrow[r, "T^{\nu}_{\Sigma'}{}^2q"] \arrow[d, shift right, "\mu_{T^{\nu}_{\Sigma'}X}"'] \arrow[d, shift left, "T^{\nu}_{\Sigma'}\mu_X"] & T^{\nu}_{\Sigma'}{}^2S \arrow[d, shift right, "\mu_S"'] \arrow[d, shift left, "T^{\nu}_{\Sigma'}\alpha"] \\
 & T^{\nu}_{\Sigma'}{}^2X \arrow[r, "T^{\nu}_{\Sigma'}q"'] \arrow[d, "\mu_X"'] & T^{\nu}_{\Sigma'}S \arrow[d, "\alpha"] \\
 & T^{\nu}_{\Sigma'}X \arrow[r, "q"'] & S
\end{tikzcd} \qquad (5.12)$$

where the top row is again a coequaliser for the same argument as above. Therefore, we will have the desired equation if we show that $\alpha\mu_S T^{\nu}_{\Sigma'}{}^2q = \alpha T^{\nu}_{\Sigma'}\alpha T^{\nu}_{\Sigma'}{}^2q$. This follows by the following chase around the diagram:

$$\begin{aligned}
\alpha\,\mu_S\,T^{\nu}_{\Sigma'}{}^2q &= \alpha\,T^{\nu}_{\Sigma'}q\,\mu_{T^{\nu}_{\Sigma'}X} \\
&= q\,\mu_X\,\mu_{T^{\nu}_{\Sigma'}X} \\
&= q\,\mu_X\,T^{\nu}_{\Sigma'}\mu_X \\
&= \alpha\,T^{\nu}_{\Sigma'}(q\,\mu_X) \\
&= \alpha\,T^{\nu}_{\Sigma'}\alpha\,T^{\nu}_{\Sigma'}{}^2q;
\end{aligned}$$

therefore $\alpha\,\mu_X = \alpha\,T^{\nu}_{\Sigma'}\alpha$ and (S,α) is a $\mathsf{T}^{\nu}_{\Sigma'}$-algebra.

We now want to show that (S, α) satisfies the equations (i.e. that $\alpha\,\lambda^\nu_S = \alpha\,\rho^\nu_S$), and that it is the free algebra on X with this property.

In order to see that the equations are satisfied, we can show that they are equal as functions between sets, since the forgetful functor $U\colon \mathsf{Pre} \longrightarrow \mathsf{Set}$ is faithful. Consider the following diagram, where on the top row $T^\nu_{E+\Sigma'}q$ is the (absolute) coequaliser of $T^\nu_{E+\Sigma'}\lambda^\nu_X$ and $T^\nu_{E+\Sigma'}\rho^\nu_X$ because of (5.10):

$$\begin{tikzcd}
T^\nu_{E+\Sigma'}{}^2X \arrow[dd, "\bar\mu_X"'] \arrow[dr, bend right=20, "(\lambda^\nu)^2_X"] \arrow[dr, bend right=35, "(\rho^\nu)^2_X"'] & T^\nu_{E+\Sigma'}T^\nu_{\Sigma'}X \arrow[l, "T^\nu_{E+\Sigma'}h_X" description] \arrow[r, "T^\nu_{E+\Sigma'}q"] \arrow[d, shift right, "\rho^\nu_{T^\nu_{\Sigma'}X}"'] \arrow[d, shift left, "\lambda^\nu_{T^\nu_{\Sigma'}X}"] \arrow[from=l, shift left=2, "T^\nu_{E+\Sigma'}\lambda^\nu_X"] \arrow[from=l, shift right=2, "T^\nu_{E+\Sigma'}\rho^\nu_X"'] & T^\nu_{E+\Sigma'}S \arrow[d, shift right, "\rho^\nu_S"'] \arrow[d, shift left, "\lambda^\nu_S"] \\
 & T^\nu_{\Sigma'}{}^2X \arrow[r, "T^\nu_{\Sigma'}q"'] \arrow[d, "\mu_X"'] & T^\nu_{\Sigma'}S \arrow[d, "\alpha"] \\
T^\nu_{E+\Sigma'}X \arrow[r, shift left, "\lambda^\nu_X"] \arrow[r, shift right, "\rho^\nu_X"'] & T^\nu_{\Sigma'}X \arrow[r, "q"] & S.
\end{tikzcd}$$

By (5.9), we can chase the diagram to show that

$$\begin{aligned}
\alpha\,\lambda^\nu_S\,T^\nu_{E+\Sigma'}q &= \alpha\,T^\nu_{\Sigma'}q\,\lambda^\nu_{T^\nu_{\Sigma'}X} \\
&= q\,\mu_X\,\lambda^\nu_{T^\nu_{\Sigma'}X} \\
&= q\,\mu_X\,\lambda^\nu_{T^\nu_{\Sigma'}X}\,T^\nu_{E+\Sigma'}(\lambda^\nu_X\,h_X) \\
&= q\,\mu_X\,(\lambda^\nu)^2_X\,T^\nu_{E+\Sigma'}h_X \\
&= q\,\lambda^\nu_X\,\bar\mu_X\,T^\nu_{E+\Sigma'}h_X \\
&= q\,\rho^\nu_X\,\bar\mu_X\,T^\nu_{E+\Sigma'}h_X \\
&= q\,\mu_X\,\rho^\nu_{T^\nu_{\Sigma'}X} \\
&= \alpha\,\rho^\nu_S\,T^\nu_{E+\Sigma'}q,
\end{aligned}$$

hence the result, since $T^\nu_{E+\Sigma'}q$ is epic.

Clearly, the composite $X \xrightarrow{\eta_X} T^{\nu}_{\Sigma'}X \xrightarrow{q} S$ gives a map $\overline{\eta}: X \longrightarrow S$. Given another $\mathsf{T}^{\nu}_{\Sigma'}$-algebra (M, β) with $\beta\lambda^{\nu}_{M} = \beta\rho^{\nu}_{M}$ and a map $\phi: X \longrightarrow M$, we get a $\mathsf{T}^{\nu}_{\Sigma'}$-algebra morphism $\widetilde{\phi}: T^{\nu}_{\Sigma'}X \longrightarrow M$ such that $\widetilde{\phi}\eta_X = \phi$, because $(T^{\nu}_{\Sigma'}X, \mu_X)$ is the free $\mathsf{T}^{\nu}_{\Sigma'}$-algebra on X:

$$\begin{tikzcd}
T^{\nu}_{\Sigma'}{}^{2}X \arrow[rr, "T^{\nu}_{\Sigma'}\widetilde{\phi}"] \arrow[d, "\mu_X"'] & & T^{\nu}_{\Sigma'}M \arrow[d, "\beta"] \\
T^{\nu}_{\Sigma'}X \arrow[rr, "\widetilde{\phi}"] & & M \\
& X \arrow[ul, bend left, "\eta_X"] \arrow[ur, bend right, "\phi"'] &
\end{tikzcd}$$

Clearly, $\widetilde{\phi} = \beta\, T^{\nu}_{\Sigma'}\phi$. We get that $\widetilde{\phi}$ factors through S if $\widetilde{\phi}\,\lambda^{\nu}_{X} = \widetilde{\phi}\,\rho^{\nu}_{X}$, and this follows by the chain of equalities:

$$\widetilde{\phi}\,\lambda^{\nu}_{X} = \beta\, T^{\nu}_{\Sigma'}\phi\,\lambda^{\nu}_{X} = \beta\,\lambda^{\nu}_{M}\,T^{\nu}_{E+\Sigma'}\phi = \beta\,\rho^{\nu}_{M}\,T^{\nu}_{E+\Sigma'}\phi = \widetilde{\phi}\rho^{\nu}_{X}.$$

We therefore get a unique morphism $\overline{\phi}: S \longrightarrow M$ such that $\overline{\phi}\,q = \widetilde{\phi}$. If we show that $\overline{\phi}$ is a $\mathsf{T}^{\nu}_{\Sigma'}$-algebra morphism, then we have proved the universal property of S. Using again the fact that $T^{\nu}_{\Sigma'}q$ is epic, this follows, since

$$\overline{\phi}\,\alpha\,T^{\nu}_{\Sigma'}q = \overline{\phi}\,q\,\mu_X = \widetilde{\phi}\,\mu_X = \beta\,T^{\nu}_{\Sigma'}\widetilde{\phi} = \beta\,T^{\nu}_{\Sigma'}\overline{\phi}\,T^{\nu}_{\Sigma'}q.$$

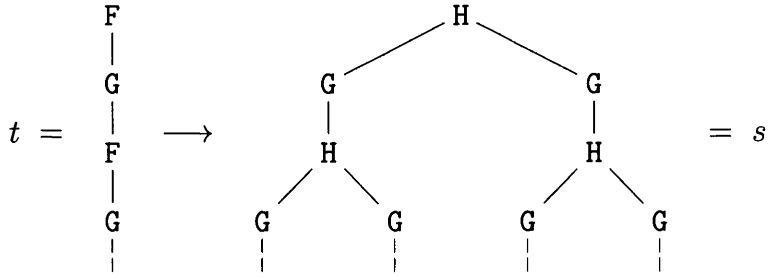
We have just proved the following.

Proposition 5.8 *The coequaliser $\mathsf{T}^{/\!/}_{\langle\Sigma,\mathcal{R}\rangle}$ of the monad morphisms λ^{ν} and ρ^{ν} is pointwise defined as the coequaliser S in (5.8).*

We can now take a closer look at the preorder $S = T^{/\!/}_{\langle\Sigma,\mathcal{R}\rangle}X$. The first thing to note is that, as for the finite case, in the $\sim$-equivalence class $\bar{t}$ of any term $t \in T^{\nu}_{\Sigma'}X$ there is precisely one term where no place-holder appears (that is the image of $\bar{t}$ under the function f defined above). Therefore, the carrier of S can be identified with the set $T^{\nu}_{\Sigma}X$ of finite and infinite terms over Σ with variables in X.

Given a term t in $T^{\nu}_{\Sigma}X$, we can *highlight* a set of disjoint redexes in it by choosing an equivalent term t' in $T^{\nu}_{\Sigma'}X$ where some l_i symbols appear. In this preorder, t' rewrites to the term s' which is obtained by replacing all the l_i's with the matching r_i's. By replacing all the r_i's with the corresponding s_i's from the rules, we get a term u and t rewrites to u in $T^{\nu}_{\Sigma}X$.

Example 5.9 Let Σ be the signature consisting of two unary symbols $\mathtt{F}$ and $\mathtt{G}$ and a binary symbol $\mathtt{H}$. Let $\mathcal{R}$ be consisting of only one rule: $\{x\} \vdash \rho\colon \mathtt{F}(x) \to \mathtt{H}(x,x)$. We want to show that in $T^{//}_{\langle\Sigma,\mathcal{R}\rangle}X$ we have the following rewrite:



(5.13)

This follows easily, since $(\mathtt{FG})^{\omega} \sim (l\mathtt{G})^{\omega}$ in $T^{\nu}_{\Sigma'}X$, where l and r are the placeholders for the rule ρ in the signature Σ'.

We then have that $(l\mathtt{G})^{\omega} \to (r\mathtt{G})^{\omega}$ in $T^{\nu}_{\Sigma'}X$, and clearly $(r\mathtt{G})^{\omega} \sim s$, when replacing $r(x)$ by $\mathtt{H}(x,x)$ at any occurrence.

The rewrite relation arising in this way is precisely that of infinite parallel rewriting as defined in [18]. The odd considerations therein about finding a sequence of finite terms approximating t and the redexes not crossing the boundaries of those terms are here totally unnecessary, since the monadic structures of $\mathsf{T}^{\nu}_{E+\Sigma'}$ and $\mathsf{T}^{\nu}_{\Sigma'}$ are taking care of the coherence properties for us. In particular, the monad $\mathsf{T}^{\nu}_{E+\Sigma'}$ provides us with the infinite proof terms for any reduction we are interested in. As in the finite case, the usual single-step rewrite relation can be recovered by highlighting precisely one redex in a term and considering the corresponding rewrite.

Notice once more how, in this framework, we do not need to restrict our attention to orthogonal TRS's in order to consider infinite parallel rewriting, since our construction selects for us only those sets of redexes which are disjoint. This gained generality is paid, of course. Namely, we had to require the rather heavy condition that our rules are non-collapsing (rules of the form $\rho: x \to t(x)$ are also banned, of course, but they are usually not considered anyway). However, this is not an unusual requirement, because of the phenomenon of collapsing towers.

5.3 Rational Rewriting

In this section, we show how the notion of rational rewriting as defined by Corradini and Gadducci [18] can also be captured by our setting. This will not require much work, and can be considered an exercise, after the previous section, since we are morally just repeating the argument on a different monad.

In their work, a rational rewrite is defined to be a parallel rewrite between rational terms where, after "tagging" all redexes, the source of the rewrite is still a rational term.

For example, in the TRS of Example 5.9 the rational term $(\mathtt{FG})^\omega$ has a redex at each occurrence of the symbol F. The corresponding tagged term $(\mathtt{F}^*\mathtt{G})^\omega$ is rational, therefore the rewrite (5.13) is a rational rewrite (see [18] for more details). If we had tagged only some occurrences of F, in such a way that the derived term was not rational, then the corresponding parallel rewrite would have not been accepted.

In our framework, tagging the redexes is the same as highlighting them. If we build rational terms (and rewrites amongst them) on the signature Σ' as defined in (5.1), we get all rational terms and only their allowed sets of highlighted redexes. Imposing equations will, as usual, produce the desired TRS where elements can be identified with the rational Σ-terms and the rewrites will be precisely the rational parallel ones. In a nutshell, we could say that we are repeating the previous section with the rational monad R for T^ν.

First of all, it has to be noted that, by Proposition 4.8, we can consider the monads $\mathsf{R}_{E+\Sigma'}$ and $\mathsf{R}_{\Sigma'}$. Assuming again that no term in the rules is just a variable, the morphisms λ' and ρ' of (5.3) are guarded. Of course, given that we want to find a parallel pair between $\mathsf{R}_{E+\Sigma'}$ and $\mathsf{R}_{\Sigma'}$, we cannot allow our rules to rewrite arbitrary infinite terms. Following the same reasoning as in Remark 5.6, we can allow the rules to involve also rational terms, instead of just the finite ones. When this is the case, we can extend the parallel pair $\lambda', \rho' \colon F_{E+\Sigma'} \longrightarrow R_{\Sigma'}$ to a pair $\lambda^{\mathrm{R}}, \rho^{\mathrm{R}} \colon \mathsf{R}_{E+\Sigma'} \longrightarrow \mathsf{R}_{\Sigma'}$ by means of the universal property of $\mathsf{R}_{E+\Sigma'}$ described in [6, Theorem 4.30].

We can then consider the coequaliser

$$\begin{tikzcd}
\mathsf{R}_{E+\Sigma'} \arrow[r, shift left, "\lambda^{\mathrm{R}}"] \arrow[r, shift right, "\rho^{\mathrm{R}}"'] & \mathsf{R}_{\Sigma'} \arrow[r, "q"] & \mathsf{T}^{\mathrm{R}}_{\langle\Sigma,\mathcal{R}\rangle}
\end{tikzcd}$$

in the category of accessible monads on **Pre**. By repeating exactly the same computations as in the previous section, we can give a description of $T^{\mathrm{R}}_{\langle\Sigma,\mathcal{R}\rangle}X$ for a preorder X. Its underlying set will be the quotient $R_{\Sigma'}X/\!\sim$, where $\sim$ is the equivalence relation generated by the image of $R_{E+\Sigma'}X$ through the pairing $\langle\lambda^{\mathrm{R}},\rho^{\mathrm{R}}\rangle$. The preorder relation on $R_{\Sigma'}X/\!\sim$ is the least making q the coequaliser in **Pre**, and it is described as before: the equivalence class of a term t rewrites to that of a term u if and only if there exist terms t' and u' with $t \sim t'$ and $u \sim u'$, such that t' and u' are equal apart from a set of nodes which in t' are labelled by l_i-symbols and in u' by the corresponding r_i's, so that t' rewrites to u' in $R_{\Sigma'}X$.

Analogously to the finite and infinite cases, in the quotient $R_{\Sigma'}X$ each equivalence class has precisely one representative which is free of place-holder symbols, i.e. which is built solely on Σ. Therefore, we can identify the carrier of $T^{\mathrm{R}}_{\langle\Sigma,\mathcal{R}\rangle}X$ with the set $R_\Sigma X$ of rational terms over Σ. Choosing a different representative for a $\sim$-equivalence class can, as usual, be interpreted as highlighting a set of disjoint redexes, but now, because terms have to be rational on Σ', we can only highlight those redexes which give rise to a rational parallel reduction. It is therefore clear that $T^{\mathrm{R}}_{\langle\Sigma,\mathcal{R}\rangle}$ is precisely the parallel rational rewrite relation defined by $\mathcal{R}$ on $R_\Sigma X$.

Remark 5.10 Notice how, once more, we are gaining for free an extension of the classical theory, in that we can build a term rewriting system over a set of variables which already have some internal rewrites (i.e. the preorder on X need not be discrete). Moreover, we do not have to rely on the system being orthogonal. As for the infinite case, we have to ask for the rules not to involve any variable, but this is our only constraint.

Some authors in the field of term rewriting who do not consider the set of terms as a complete partial order (usually because they prefer stressing their metric properties) manage to allow at most one collapsing rule, by introducing a dummy function symbol whose purpose is that of making the equation non collapsing again. So, for example, the rule $\rho\colon \mathtt{A}(x) \to x$ which we came across before would become $\rho\colon \mathtt{A}(x) \to \varepsilon(x)$ for a new symbol ε. In this case, the (previously) collapsing tower $\mathtt{A}^\omega$ rewrites in parallel to the term ε^ω, which we can interpret as a meaningless term.

We feel that our formulation provides a cleaner semantics to the notion of parallel rewriting, as well as being a nice application of the results presented in Chapter 3 and 4. Given other classes of terms or syntactic-like structures by means of monads on Set, it would be interesting to see if they admit an extension to Pre which could allow to develop a rewriting theory for them.

From a purely categorical point of view, this chapter presents a rather atypical situation, where a coequaliser of monads can be computed pointwise. At present, we cannot see any way of generalising this example to a wider setting.

Chapter 6

Conclusions and Further Work

The purpose of this thesis was to give a categorical account of syntactical structures with an infinitary character. The motivations for this kind of research come from the practice of dealing with infinite terms in both mathematics and computer science. The paradigmatic example is that of trying to give a semantics to infinite computations of a program. Syntactic models of infinite structures or infinite behaviours rely on some form of completion on the sets of terms, so that we can describe an infinite term by means of its finite approximations, and likewise we can identify an infinite computation by its finite observations. The required completeness is achieved by either considering an ordering or a metric on the terms. Adámek showed [4] how this is just a reflection of the fact that finite and infinite terms form respectively the initial and the final coalgebra for the same endofunctor of an lfp category. This is not the only aspect in which finite and infinite terms are dual. The logics used to prove properties of them are also dual [40]. For all these reasons, it seems reasonable to use categorical methods in order to exploit this duality and fit the different elements into a unique framework.

The other determining motivation for introducing the categorical notation was that it allows us to unify a wide variety of concepts which are usually treated separately. The starting point in this direction is the paper by Kelly and Power [37], which we cited several times throughout the thesis. The idea of encoding various structures as equational theories over some signature has proved very fruitful, and captured multi-sorted theories, categories with structures as well as several different forms of rewriting. In their work, they start with a finitary signature and they build the free monad over it, by pointwise considering the initial algebra, thus providing a semantic for the given equational theory.

In order to provide a semantic for infinite terms, it was therefore natural to explore possible dualisations of their work. This gave rise to the situations described in Table 2.1.

6.1 Achievements

Of the three possible dualisations which we found, we explored only two, because we found them more meaningful. In both cases we could not recover the whole picture. When considering cofree comonads on cosignatures, we had no handle on the rank of our structures (although the work of Worrell [58] might suggest some form of upper bound to the increase of the rank). When considering monads of infinite terms, we can impose equations, but we tend to loose the relevant properties of our monads in the coequaliser (the coequaliser of two free completely iterative monads is not the free completely iterative monad representing the corresponding equational theory).

However, we hope to have convinced the reader that the proposed dualisations are of interest and are worth some further investigation. Coalgebras are nowadays widely used in the specification of behaviours, and a great amount of research has been published on the logic underneath them. Our comonadic treatment fits in this framework, by giving a very intuitive approach to the notions of cosignature and coequations. This bridges the gap towards the several notions of co-Birkhoff theorem which have appeared recently in the literature [11].

When considering a monadic structure on final coalgebras, though, we felt that we had hit the right point. Despite some initial suspicion by the audience when we first conjectured the existence of the monad T^{ν}, within a couple of years three different proofs were given of this result [50, 2, 27]. The importance of the result stands in its wide applicability and in the universal property of T^{ν}, which Adámek and his coauthors proved. However, the totality of infinite terms is excessive for the common practice, because we cannot even describe them. For this reason all the aforementioned research groups turned their attention to other classes of terms, such as the rational ones. Theorem 3.16 was our way of describing monads of suitable classes of terms or syntactic structures in general. The possibility of unifying the description of rational terms and term graphs is our reason for introducing it. The notion of (strongly) guarded monad also proved fruitful in that we can solve algebraic systems of equations over them, and the set of algebraic terms is the natural successive step, after the rational ones, in the chain going from finite to the infinite terms.

Although we did not develop a general enriched theory of strongly guarded monads, we could certainly handle the Pre-enrichment "on the fly", and a simple instantiation of our results turned out to describe precisely some well-known notions in the theory of term rewriting systems.

6.2 Further Directions of Research

Finding what looked like a reasonable point where to stop the research in order to give to this thesis a sense of accomplishment is certainly one of the most arbitrary choices we have taken. In fact, as it is natural to expect with all new research, there are several threads which can be explored. We chose to hide them as much as possible during the exposition, in order not to distract the reader from those which instead we brought to an end, but each of them would deserve a research on its own.

The most urgent question is that of considering infinite equational theories. Of course, the problem is not just categorical. Given a signature Σ and a set E of equations, it is not clear what the free (completely) iterative theory satisfying the equations should be. Of course we can form a coequaliser like we did in Chapter 5, but this may not be exactly what we want. In the set of finite and infinite Σ-terms any guarded system of equations has a unique solution, but this may be no longer valid if we consider its quotient along the equations. In other terms, it is not clear how to give a completely iterative monad T such that TX is the free iterative theory on $\langle \Sigma, E \rangle$.

The problem is more significant than it might look at first sight, and seems to be intrinsically related to that of defining a monad of algebraic terms. Finding a way of solving algebraic systems of equations was a first step towards a solution, but in the months following that result we could not finalise our efforts to define such a monad. The problem is of interest not just to us, and we often discussed the matter with the Braunschweig group, but none of us succeeded, so far. The link with the problem of considering equations on rational terms is the explicit substitution monad. Given a signature Σ, one can enrich it by adding symbols for explicit substitution, which take a term t on n variables and terms $t_1, \ldots, t_n$ to give a new term which represents (but is not!) the substitution of the t_i's for the variables in t. Courcelle [19] proved that any algebraic term can be obtained from a rational term over this enhanced signature, by removing the explicit substitution symbols and performing the substitutions which they stand for. Evaluating the explicit substitution symbols is somehow like imposing equations on the rational monad over the enhanced signature, and we know how to introduce the explicit substitution symbols into our context; therefore, we conjecture that a solution to the first problem would yield a natural solution to this as well.

In Chapter 5, we gave a sound semantics for the notion of parallel rewriting in the finite, infinite and rational cases. As we mentioned, though, infinite parallel rewriting is not the most preferable form of infinite rewriting, and many others have been introduced. In particular, strongly convergent rewriting [38] seems to be quite fitting for modelling the computation of λ-calculus. A categorical semantics to such a notion would clearly be valuable.

Rational terms form the free iterat*ive* theory on a signature, i.e. the free one where each guarded system of equations has a unique solution. A similar, yet different notion is that of an iterat*ion* theory [15], where *any* system (regardless of being guarded) has at least one solution (possibly many more). A categorical model of such theories has been given by Plotkin and Simpson [54]. It is clearly worth investigating the connections between the two.

Recursive equations appear in all areas of mathematics and computer science, and fixpoint theorems are always sought and used thoroughly. Having generalised the construction of the rational monad to all lfp categories (and even to monoidal categories, if necessary [6]), we should be able to recover well-known structures with an infinitary flavour. For example, we know that the set of polynomials over a ring can be presented as a free monoid over the object of the variables which the polynomials are built on. The formal power series, are easily seen to be the set of infinite terms for that same theory, so what about the rational terms? We know that a term like $1 + X + X^2 + X^3 + \dots$ is the inverse (in the set of formal power series) of the polynomial $1 - X$. But not all invertible finite polynomials give rise to a rational term. Our conjecture is that they actually form the algebraic terms, but one should first be able to give a categorical description of them!

Finally, more speculatively, we could investigate whether our machinery allows us to reason about geometric figures which present some recursive pattern. In fact, self-similar fractals [23] are fixpoints for a contractive map on the compact subspaces of some $\mathbb{R}^n$, and this suggests some connection.

Bibliography

[1] Peter Aczel, Jiří Adámek, Stefan Milius, and Jiří Velebil. Infinite trees and completely iterative theories: a coalgebraic view. *Theoretical Computer Science*, 300:1–45, 2003.

[2] Peter Aczel, Jiří Adámek, and Jiří Velebil. A coalgebraic view of infinite trees and iteration. In Andrea Corradini, Marina Lenisa, and Ugo Montanari, editors, *Proceedings 4th Workshop on Coalgebraic Methods in Computer Science, CMCS'01, Genova, Italy, 6–7 Apr. 2001*, volume 44(1) of *Electronic Notes in Theoretical Computer Science.* Elsevier, Amsterdam, 2001.

[3] Jiří Adámek. Free algebras and automata realizations in the language of categories. *Commentationes MathematicæIniversitatis Carolinæ*, 15:589–602, 1974.

[4] Jiří Adámek. Final coalgebras are ideal completions of initial algebras. *Journal of Logic and Computation*, 12(2):217–242, 2002.

[5] Jiří Adámek. On final coalgebras of continuous functors. *Theoretical Computer Science*, 294, 2003.

[6] Jiří Adámek, Stefan Milius, and Jiří Velebil. Free iterative theories: a coalgebraic view. *Mathematical Structures in Computer Science*, 13:259–320, 2003.

[7] Jiří Adámek and Hans-E. Porst. On tree coalgebras and coalgebra presentations. submitted.

[8] Jiří Adámek and Jiří Rosicky. *Locally Presentable and Accessible Categories*, volume 189 of *London Mathematical Society Lecture Notes.* Cambridge University Press, 1994.

[9] Jiří Adámek and Vera Trnková. *Automata and algebras in categories.* Kluwer, 1990.

[10] A. Arnold and M. Nivat. The metric space of infinite trees. algebraic and topological properties. *Fundamenta Informaticae*, 3(4):181–205, 1980.

[11] Steve Awodey and Jesse Hughes. The coalgebraic dual of Birkhoff's variety theorem. Unpublished manuscript, October 2000.

[12] Michael Barr. Coequalizers and free triples. *Math. Z.*, 116:307–322, 1970.

[13] Michael Barr. Terminal coalgebras for endofunctors on sets. available from `ftp://www.math.mcgill.ca/pub/barr/trmclgps.zip`, 1999.

[14] Michael Barr and Charles Wells. *Toposes, Triples and Theories.* Springer Verlag, 1985.

[15] Stephen L. Bloom and Zoltán Ésik. *Iteration Theories: The Equational Logic of Iterative Processes.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, Berlin, 1993.

[16] Francis Borceux. *Handbook of Categorical Algebra-I.* Number 50 in Encyclopedia of Mathematics. Cambridge University Press, 1994.

[17] Corina Cirstea. An algebra-coalgebra framework for system specification. In *Proceedings of CMCS'00*, volume 33 of *Electronic Notes in Theoretical Computer Science.* Elsevier, 2000.

[18] Andrea Corradini and Fabio Gadducci. Rational term rewriting. *Lecture Notes in Computer Science*, 1378:156–171, 1998.

[19] Bruno Courcelle. Fundamental properties of infinite trees. *Theoretical Computer Science*, 25(2):95–169, 1983.

[20] Eduardo J. Dubuc and Gregory Max Kelly. A presentation of topoi as algebraic relative to categories or graphs. *Journal for Algebra*, 81:420–33, 1983.

[21] Calvin Elgot. Monadic computations and iterative algebraic theories. In H. E. Rose and John C. Shepardson, editors, *Logic Colloquium '73*, pages 175–239. North-Holland, 1975. Report RC-4564, IBM T.J. Watson Research Center, Yorktown Heights NY, October 1973.

[22] Calvin C. Elgot, Stephen Bloom, and Ralph Tindell. On the algebraic structure of rooted trees. *Journal of Computer and System Sciences*, 16(3):362–399, June 1978.

[23] K. J. Falconer. *Fractal geometry : mathematical foundations and applications.* Chichester ; New York : Wiley, 1990, 288 p. CALL NUM: QA614.86 .F35 1990, 1990.

[24] Marcelo Fiore, Gordon Plotkin, and Daniele Turi. Abstract syntax and variable binding. In G. Longo, editor, *Proceedings of the 14th Annual Symposium on Logic in Computer Science (LICS'99)*, pages 193–202, Trento, Italy, 1999. IEEE Computer Society Press.

[25] Neil Ghani, Cristoph Lüth, and Federico De Marchi. Coalgebraic approaches to algebraic terms. presented at FICS 2002 - Fixed Points in Computer Science, 20-21 July 2002, Copenhagen, Denmark.

[26] Neil Ghani, Cristoph Lüth, and Federico De Marchi. Coalgebraic monads. In Lawrence M. Moss, editor, *Proceedings 5th Workshop on Coalgebraic Methods in Computer Science, CMCS'02, Grenoble, France, 6–7 Apr. 2002*, 2002.

[27] Neil Ghani, Cristoph Lüth, Federico De Marchi, and John Power. Algebras, coalgebras, monads and comonads. In Andrea Corradini, Marina Lenisa, and Ugo Montanari, editors, *Proceedings 4th Workshop on Coalgebraic Methods in Computer Science, CMCS'01, Genova, Italy, 6–7 Apr. 2001*, volume 44(1) of *Electronic Notes in Theoretical Computer Science.* Elsevier, Amsterdam, 2001.

[28] Susanna Ginali. Regular trees and the free iterative theory. *Journal of Computer and System Sciences*, 18(3):228–242, June 1979.

[29] J. A. Goguen, J. W. Thatcher, E. G. Wagner, and J. B. Wright. Initial algebra semantics and continuous algebras. *Journal of the ACM*, 24(1):68–95, January 1977.

[30] Graham Hutton and Jeremy Gibbons. The generic approximation lemma. To appear in *Information Processing Letters*, 2001.

[31] Bart Jacobs and Jan Rutten. A tutorial on (co)algebras and (co)induction. *Bulletin of the EATCS*, 62:222–259, 1996.

[32] Peter T. Johnstone. *Stone spaces*, volume 3 of *Cambridge studies in advanced mathematics*. Cambridge University Press, 1982.

[33] Peter T. Johnstone, A. John Power, T. Tsujishita, H. Watanabe, and James Worrell. An axiomatics for categories of transition systems as coalgebras. In *Proc. Thirteenth IEEE Symp. on Logic in Computer Science*. IEEE Computer Soc. Press, 1998.

[34] Gregory Max Kelly. A unified treatment of transfinite constructions. *Bull. of Austral. Math. Soc.*, 22:1–83, 1980.

[35] Gregory Max Kelly. *Basic Concepts of Enriched Category Theory*, volume 64 of *London Mathematical Society Lecture Notes*. Cambridge University Press, 1982.

[36] Gregory Max Kelly. Structures defined by finite limits in the enriched context, i. *Cahiers Topologie Géom. Différentielle*, 23:3–42, 1982.

[37] Gregory Max Kelly and John Power. Adjunctions whose counits are equalizers, and presentations of finitary monads. *Journal of Pure and Applied Algebra*, 89:163–179, 1993.

[38] J.R. Kennaway and F.J. de Vries. Infinitary rewriting. In Terese, editor, *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*, pages 668–711. Cambridge University Press, 2003.

[39] J.W. Klop. Term rewriting systems. In *Handbook of Logic in Computer Science, Volumes 1 (Background: Mathematical Structures) and 2 (Background: Computational Structures), Abramsky & Gabbay & Maibaum (Eds.), Clarendon*, volume 2. 1992.

[40] A. Kurz. Logics for coalgebra and applications to computer science, 2000. Dissertation, Ludwig-Maximilians-Universität München.

[41] J. Lambek. Subequalizers. *Canadian Math. Bull.*, 13:337–349, 1970.

[42] C. Lüth. *Categorical Term Rewriting: Monads and Modularity.* PhD thesis, University of Edinburgh, 1997.

[43] Christoph Lüth and Neil Ghani. Monads and modular term rewriting. In E. Moggi and G. Rosolini, editors, *Proceedings 7th Int. Conf. on Category Theory and Computer Science, CTCS'97, Santa Margherita Ligure, Italy, 4–6 Sept. 1997*, volume 1290 of *Lecture Notes in Computer Science*, pages 69–86. Springer-Verlag, Berlin, 1997.

[44] Saunders Mac Lane. *Categories for the Working Mathematician.* Springer Verlag, 1971.

[45] Federico De Marchi, Neil Ghani, Cristoph Lüth, and John Power. Dualising initial algebras. *Mathematical Structures in Computer Science*, 13:349–370, 2003.

[46] Stefan Milius. Free iterative theories: a coalgebraic view (extended abstract). presented at FICS 2001 - Fixed Points in Computer Science, 7-8 September 2001, Florence, Italy.

[47] Robin Milner. *A Calculus for Communicating Systems*, volume 92 of *Lecture Notes in Computer Science.* Springer-Verlag, Berlin, 1980.

[48] Lawrence Moss. Parametric corecursion. preprint, available at `http://math.indiana.edu/home/moss/parametric.ps`.

[49] Lawrence Moss. Recursion and corecursion have the same equational logic. preprint, available at `http://math.indiana.edu/home/moss/eqcoeq.ps`.

[50] Lawrence S. Moss. Coalgebraic logic. *Annals of Pure and Applied Logic*, 96(1–3):277–317, 1999.

[51] Edmund Robinson. Variations on algebra: monadicity and generalisation of equational theories. Technical Report 6/94, Sussex Computer Science, 1994.

[52] J. J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000.

[53] D. E. Rydeheard and J. G. Stell. Foundations of equational deduction: a categorical treatment of equational proof and unification algorithms. *Category Theory and Computer Science*, 283:114–139, 1987.

[54] Alex Simpson and Gordon Plotkin. Complete axioms for categorical fixed-point operators. In *Proceedings 15th Annual IEEE Symp. on Logic*

in Computer Science, LICS'00, S.ta Barbara, CA, USA, 26–29 June 2000, pages 30–44. IEEE Computer Society Press, Los Alamitos, CA, 2000.

[55] Ross Street. The formal theory of monads. *Journal of Pure and Applied Algebra*, 2:149–168, 1972.

[56] Daniele Turi. *Functorial Operational Semantics and Its Denotational Dual.* PhD thesis, Free University, Amsterdam, 1996.

[57] Daniele Turi and Jan Rutten. On the foundations of final coalgebra semantics: non-well-founded sets, partial orders, metric spaces. *Mathematical Structures in Computer Science*, 8(5):481–540, October 1998.

[58] James Worrell. Terminal sequences for accessible endofunctors. In *Proceedings of CMCS '99*, number 19 in Electronic Notes in Theoretical Computer Science, 1999.

Index