Von der Carl-Friedrich-Gauß-Fakultät für Mathematik
und Informatik der Technischen Universität Braunschweig

genehmigte Dissertation

zur Erlangung des Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

Stefan Milius

Coalgebras, Monads and Semantics

28. Oktober 2005

1. Referent: Prof. Jiří Adámek, Technische Universität Braunschweig, Deutschland

2. Referent: Prof. Lawrence S. Moss, Indiana University, Bloomington, Indiana, USA

eingereicht am: 14. Juli 2005

To the memory of my mother.

LEBENSLAUF

ZUR PERSON

Name	Stefan Milius
Geburtsdatum	3. Juni 1975
Geburtsort	Magdeburg
Familienstand	verheiratet, zwei Kinder
Email	milius@iti.cs.tu-bs.de
WWW	www.iti.cs.tu-bs.de/∼milius

AUSBILDUNG UND BISHERIGE TÄTIGKEITEN

08/1990 – 07/1994	Werner-von-Siemens-Gymnasium in Magdeburg (bis 1992 Spezialschule mathematisch-naturwissenschaftlich-technischer Richtung) Abschluss: Abitur
07/1994 – 09/1995	Zivildienst bei den Evangelischen Stiftungen Neuerkerode
10/1995 – 09/2000	Informatikstudium an der TU Braunschweig Abschluss: Diplom
9/1999 – 07/2000	Studium der Mathematik an der York University, Toronto, Ontario, Canada Abschluss: Master of Arts
seit 9/2000	wissenschaftlicher Mitarbeiter am Institut für Theoretische Informatik der TU Braunschweig
10/2005	Promotion am Institut für Theoretische Informatik der TU Braunschweig Mentor: Prof. Jiří Adámek Abschluss: Dr. rer. nat.

AUSZEICHNUNGEN

Mitglied der Studienstiftung des deutschen Volkes	02/1998 – 09/2000
TU Preis für hervorragende Leistungen im Studium	10/2000

About this Thesis

This is a cumulative dissertation which consists of the following papers:

[AMV_1]	Free Iterative Theories: a coalgebraic view
[AMV_2]	From Iterative Algebras to Iterative Theories
[AMV_3]	Elgot Algebras
[AAMV]	Infinite Trees and Completely Iterative Theories: A Coalgebraic View
[M_1]	Completely Iterative Algebras and Completely Iterative Monads
[M_2]	On Iteratable Endofunctors
[MM]	The Category Theoretic Solution of Recursive Program Schemes

These papers contain some of the results of my research obtained while I was working at the Institute of Theoretical Computer Science of the Technical University of Braunschweig, Germany, under the supervision of Professor Jiří Adámek. They have all been published in international journals and/or they have been presented at international conferences.

The current document constitutes an introduction and summary of this research. As such it contains (almost) no new results as compared to the above papers. Its aim is to provide the reader with a self-contained account of our research in a unified notation and presentation. The current document is structured in a way so as to maximize accessibility to the reader while still providing enough explanation of the techniques and tools employed in our work. All important results from the above papers are included in this document. However, in a summary, one is forced to omit material. And so many of the technical details, and certainly all the technical calculations have been omitted so that full proofs are almost never given here. We provide sketches of proofs meant to illustrate the main ideas at work in our theory. For those readers who want to delve deeper into the omitted details we have annotated our results with precise references to their places in the above papers, which we include in an appendix.

Abstract

We study mathematical structures arising in the theory of *coalgebras* which are useful for providing semantics of recursive specifications. We begin by investigating (completely) iterative algebras and (complete) Elgot algebras, and we establish the connection of these algebras to coalgebra—final coalgebras are precisely the same as free completely iterative algebras, or free complete Elgot algebras, respectively. Similarly, for the non-complete case we show that free iterative algebras, and free Elgot algebras, respectively, exist and can be constructed from finite coalgebras. Furthermore, we show that the *monads* arising from free (completely) iterative algebras are characterized by a universal property; they are free (completely) iterative monads. This generalizes and extends classical work of Calvin Elgot and Evelyn Nelson. The second main topic of this thesis is to provide a category theoretic *semantics* of so-called recursive program schemes. By exploiting the structure of free completely iterative monads and of complete Elgot algebras we are able to provide an uninterpreted as well as an interpreted semantics of recursive program schemes. We show that both are connected as expected from the classical work. Finally, we present applications of our abstract theory. We demonstrate how to recover the usual denotational semantics using ordered or metrized structures, and we present new applications defining for example operations satisfying certain equations, or fractals recursively.

Zusammenfassung

In dieser Arbeit werden mathematische Strukturen untersucht, die in der Theorie der *Koalgebren* auftauchen und welche Anwendung für die Semantik rekursiver Definitionen finden. Zunächst werden dazu (vollständig) iterative Algebren und (vollständige) Elgot Algebren betrachtet, und es wird deren Verbindung zu Koalgebren offengelegt – finale Koalgebren sind genau die freien vollständig iterativen Algebren bzw. die freien vollständigen Elgot Algebren. Analog hierzu zeigen wir, dass freie (nicht notwendig vollständige) iterative Algebren bzw. freie Elgot Algebren existieren und in kanonischer Weise aus endlichen Koalgebren konstruiert werden können. Weiterhin wird bewiesen, dass die *Monaden*, welche sich aus freien (vollständig) iterativen Algebren ergeben, durch eine universelle Eigenschaft charakterisiert sind; sie sind die freien (vollständig) iterativen Monaden. Dieser Teil der Arbeit verallgemeinert und erweitert klassische Ergebnisse von Calvin Elgot und Evelyn Nelson. Das zweite Hauptthema der vorliegenden Dissertation ist eine kategorientheoretische *Semantik* so genannter rekursiver Programschemata. Durch die Ausnutzung der Struktur freier vollständig iterativer Monaden und vollständiger Elgot Algebren sind wir in der Lage sowohl eine uninterpretierte als auch eine interpretierte Semantik rekursiver Programmschemata anzugeben. Wir zeigen auch, dass zwischen beiden Semantiken ein präziser Zusammenhang besteht, wie es auch aus den klassischen Arbeiten zu erwarten wäre. Zuletzt präsentieren wir noch einige Anwendungen unserer abstrakten Theorie. Die übliche denotationelle Semantik, welche geordnete oder metrisierte Strukturen benutzt, ergibt sich als Spezialfall aus unseren Ergebnissen. Darüberhinaus geben wir neue Anwendungen an, die zum Beispiel rekursive Definitionen von Operationen, welche gewisse Gleichungen erfüllen, oder von Fraktalen betreffen.

Acknowledgments

I would like to express my deepest gratitude to my supervisor Jiří Adámek. Without his constant help, support and encouragement the results in this thesis could never have been developed. I am very grateful for the opportunity to work with him, for all he has done to help promote my research, for example by giving me the chance to present it at numerous conferences, and to meet many great scientists.

My warmest thanks also go to Jiří Velebil, with whom I very closely collaborated, for the countless discussions, the numerous diagram chasing sessions and for always suggesting to continue our research and not to get lost in the everyday tasks that so easily get you distracted.

I also give my best thanks to Larry Moss. His work was of great inspiration to this research and our discussion in Bloomington while I was visiting for a summer school let the pieces of the puzzle concerning algebraic semantics fall into their correct place for me. The subsequent collaboration was very fruitful and of great pleasure for me.

Heartfelt thanks go to all the people who have influenced this research with their work, who expressed encouragement and with whom I had the pleasure to discuss or share ideas: Peter Aczel, Stephen Bloom, Dominique Bourn, Neil Ghani, Tom Leinster, Federico De Marchi, Alex Simpson, Dana Scott, Tarmo Uustalu, to name just a few.

Many thanks also have to go to my colleagues at the Institute of Theoretical Computer Science for all their help and support: Katja Barkowsky, Jürgen Koslowski, Frank Rust and Dietmar Wätjen.

Last but not least, I would like to extend my gratitude to my friends and family; my parents, my children and especially to my loving wife Birgit. Her belief in me and her never ending support have often helped me a great deal through the ups and downs during the last five years.

This text has been typeset using Leslie Lamport's LaTeX and the diagrams were drawn using the XY-pic macros of Kristoffer H. Rose and Ross Moore.

Contents

A youth who had begun to read geometry with Euclid, when he had learnt the first proposition, inquired, "What do I get by learning these things?" So Euclid called a slave and said "Give him threepence, since he must make a gain out of what he learns."

Joannes Stobaeus, Four Books of Extracts, Sayings and Precepts (Extracts), 5th century AD.

1 Introduction

The greatest challenge to any thinker is
stating the problem in a way that will allow a solution.
Bertrand Russell.

Recursion arises everywhere in computer science. It allows for a clear, concise and finite description of complicated or potentially infinite objects or phenomena. Hence, it is a major ingredient of any programming language or formalism describing the work of an information system. For example, a programmer often defines a function using a recursive definition; that means a definition that is self-referential—the result of the function at an argument is given by computing the same function at a modification or on parts of the given argument. The paradigm example is the recursive definition of the factorial function:

$$f(n) := \mathsf{if}\ n = 0\ \mathsf{then}\ 1\ \mathsf{else}\ n \cdot f(n-1)\,. \tag{1.1}$$

In process algebra one uses recursion to specify repeated or potentially infinite system behaviour by finite means. A very basic example is a process term

$$P := a.P \tag{1.2}$$

describing a system which can perform the action a repeatedly without ever terminating. Yet another example are recursive specifications of abstract data types such as

$$\mathsf{list} ::= \mathsf{element} \mid \mathsf{element.list} \tag{1.3}$$

where element is some given data type and list is the recursively specified datatype of lists with entries of type element.

Of course, every recursive definition or self-referential specification immediately raises the question what its meaning should be. It is the task of *semantics*, an important branch of theoretical computer science, to answer such questions.

There are different ways to give semantics to a recursive definition. In practise, it is often sufficient to consider a so-called operational semantics, which gives a meaning by running a recursive program on an abstract machine of some sort. For example, an interpreter of a functional programming language that can run a program such as (1.1) provides its operational semantics. In process algebra, one uses the so-called structured operational semantics to provide a labelled transition system that behaves as specified by a process term such as (1.2). Any actual implementation of a data type of lists containing entries of a data type of elements in a programming language gives an operational semantics to the recursive data type specification (1.3).

In this thesis we shall be concerned with another flavour of semantics—denotational semantics—which assigns to a recursive definition a certain mathematical object, its meaning. Such a semantics is independent of a particular implementation of a program, and so it more easily serves as a basis for formal reasoning about recursive programs or recursively specified information systems.

The most common approach to denotational semantics considers ordered structures in which it is possible to obtain the semantics of some recursive definition as a join of its finite approximations. Another approach considers structures equipped with a complete metric. Here it is possible to obtain semantics of recursive definitions using a limit of a Cauchy sequence of its finite approximations. In both approaches fixed points of certain functions play a major rôle. In fact, in ordered structures one usually employs the fixed point theorem of Alfred Tarski, Stephen Kleene, and Bronislaw Knaster, see e.g. [Ta]. And in completely metrized structures one applies the fixed point theorem of Stefan Banach, see [Ban].

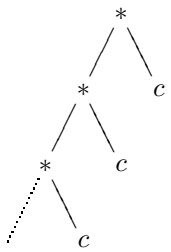
The starting point of our research is the classical work of Calvin Elgot and his collaborators [E, BE, EBT]. Their aim was to study the semantics of recursive computations at a purely algebraic level working without ordered or metrized structures. In [E] Elgot introduced *iterative theories*, which are algebraic theories in the sense of William Lawvere [La] that admit unique solutions of certain recursive specifications. Let us illustrate this on the simple theories given by operations without any equations. Suppose we have a signature Σ of operation symbols with prescribed arities in the set of natural numbers, i. e., $\Sigma = (\Sigma_n)_{n\in\mathbb{N}}$ is a sequence of sets, where Σ_n is the set of operation symbols of arity n. For example, a signature with a constant symbol c and a binary operation symbol $*$ has $\Sigma_0 = \{\, c \,\}$, $\Sigma_2 = \{\, * \,\}$, and $\Sigma_n = \emptyset$ else. For a given set of generators one can form Σ-terms over that set. Now consider a recursive system of formal equations

$$\begin{aligned} x_0 &\approx t_0(x_0, \ldots, x_n, y_0, \ldots, y_k) \\ &\vdots \\ x_n &\approx t_n(x_0, \ldots, x_n, y_0, \ldots, y_k) \end{aligned} \tag{1.4}$$

where $X = \{ x_0, \ldots, x_n \}$ is a finite set of variables, $Y = \{ y_0, \ldots, y_k \}$ is a set of parameters, and the t_i, $i = 1, \ldots, n$, are Σ-terms over the disjoint union of X and Y. Iterative theories are defined by the property that every *guarded* system (1.4) can be solved uniquely. Guardedness here means that no term t_i, $i = 1, \ldots, n$, is simply a single variable. One important example of an iterative theory is the theory T_Σ formed by all (finite and infinite) Σ-trees, i. e., rooted and ordered trees, where each inner node with n children is labelled by an n-ary operation symbol from Σ and leaves are either labelled by a constant symbol from Σ_0 or by an element from a set of generators. For example, for the signature with a constant symbol c and a binary symbol $*$ we can uniquely solve the formal equation

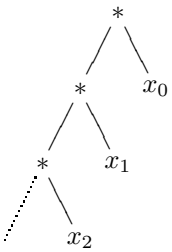
$$x \approx x * c$$

whose solution is, of course, the infinite Σ-tree



(1.5)

Another important iterative theory is the theory R_Σ, the subtheory of T_Σ formed by all *rational* Σ-trees, i. e., those Σ-trees which have (up to isomorphism) only a finite set of different subtrees—a description provided by Susanna Ginali [Gi$_2$]. For example the tree (1.5) is rational since its only subtree besides itself is the single-node tree labelled by c. But the tree



over an infinite set of generators containing all x_i, $i \in \mathbb{N}$, is not rational.

Rational trees are equivalently described as that class of Σ-trees arising as unfoldings of all guarded formal recursive equations (1.4) as infinite trees, see e.g. [C]. In fact, Calvin Elgot, Stephen Bloom and Ralph Tindell [EBT] proved that rational Σ-trees form the free iterative theory on a given signature. Similarly, the theory T_Σ of all Σ-trees is a free *completely* iterative theory on Σ, where completeness refers to the fact that all (not necessarily finite) systems of formal equations can be uniquely solved.

Algebraic theories are complicated objects to study and in fact, the original proof of Elgot and his coauthors is spread out over more than a hundred pages in the three papers [E, BE, EBT]. So it was a great step forward when Evelyn Nelson [N] and Jerzy Tiuryn [T] introduced iterative algebras for a signature to obtain a more easy approach to iterative theories. Recall first that an algebra for a signature Σ is a set A equipped with operations $\sigma_A : A^n \longrightarrow A$ for all operation symbols $\sigma \in \Sigma_n$, $n \in \mathbb{N}$. An iterative algebra A is a Σ-algebra in which every system of formal equations (1.4) has (for every interpretation of the parameters $y_0, \ldots, y_k$ in A) a unique solution, i. e., there exists a unique n-tuple of elements of A that when plugged in for the variables on both sides of the equation turn the formal equations into actual identities. Classical algebras like groups or lattices are usually not iterative. However, there are enough interesting examples of iterative algebras. Denote by $T_\Sigma X$ the algebra of all Σ-trees over a set X of generators, i. e., Σ-trees whose leaves are labelled by constant symbols from Σ_0 or generators from X. Similarly, $R_\Sigma X$ is the algebra of rational Σ-trees over X. Both $T_\Sigma X$ and $R_\Sigma X$ are iterative algebras. Moreover, Evelyn Nelson proved that $R_\Sigma X$ is a free iterative algebra on X, and that the algebraic theory of free iterative algebras is the free iterative theory of Calvin Elgot.

The first, more mathematical part, of our work presented here concerns a generalization and extension of the classical work of Calvin Elgot and Evelyn Nelson using ideas from the theory of *coalgebras*. Coalgebras have only recently gained more attention from researchers in theoretical computer science. They allow a

uniform investigation of state-based systems of different types at an abstract level, and they provide general methods for reasoning about systems. For example, sequential automata can be captured as coalgebras and, most prominently, labelled transition systems are coalgebras. Coalgebras are best studied using the language of category theory; the type of a class of systems to be captured as coalgebras is described by an endofunctor of Set, the category of sets and functions. For example, labelled transition systems are the coalgebras for the endofunctor $\mathcal{P}(\mathsf{Act} \times \_)$, where Act is a fixed set of actions and $\mathcal{P}(\_)$ denotes the power set functor. A very important concept in the theory of coalgebras is that of a final coalgebra. Final coalgebras provide in many cases a semantics of behaviour of a state of a system. For example, for automata considered as coalgebras the final coalgebra consists of all formal languages over the input alphabet, and the finality principle assigns to every state of a given automaton the language accepted by it with the chosen state as the initial one. Unfortunately, for arbitrary labelled transition systems there can be no final coalgebra for cardinality reasons. However, for finitely branching labelled transition systems the final coalgebra exists and consists of (bisimilarity equivalence classes of) all possible behaviours of states of such transition systems. These behaviours can be described as equivalence classes of finitely branching extensional trees. For more details on this and other examples we refer the reader to our introductory Section 2.3.

For our work, a crucial observation is that for a signature Σ the set $T_\Sigma X$ of all Σ-trees over X is a final coalgebra for the endofunctor $H_\Sigma(\_) + X$, where H_Σ is the canonical polynomial endofunctor associated to the signature Σ and $+$ denotes disjoint union of sets. The important message of our results is that the finality principle is sufficient to prove the classical results of Calvin Elgot. Furthermore, the fact that we work in a category theoretic setting using universal properties such as finality allows us to substantially generalize the classical results. Our proof is shorter than the classical one given by Elgot et al., and we believe that it unveils what mechanisms really are at work in the classical setting at a conceptual level.

Our overall assumptions are small indeed; we work with any endofunctor H of Set (or a more general category satisfying some rather mild side conditions) which has "enough final coalgebras", i. e., for every set X there exists a final coalgebra for $H(\_) + X$. In the first part of our work we introduce completely iterative algebras for an endofunctor, where systems like (1.4), but not necessarily with a finite set of variables, have a unique solution. For a signature Σ, every algebra $T_\Sigma X$ of Σ-trees over a set X is completely iterative. Other examples stem from the realm of algebras on complete metric spaces. For example, the golden ratio φ can be described by the continued fraction

$$1 + \cfrac{1}{1 + \cfrac{1}{1 + \cdots}}\,,$$

whence it is the unique real number from the interval $[1, 2]$ satisfying the equation

$$x = 1 + \frac{1}{x}\,.$$

Another example is the famous Cantor space c which is the unique non-empty closed subspace of the interval $[0, 1]$ such that the equation

$$c = \frac{1}{3}c + \left(\frac{2}{3} + \frac{1}{3}c\right)\,, \tag{1.6}$$

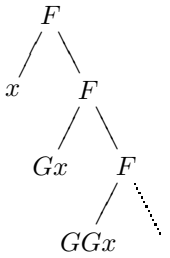
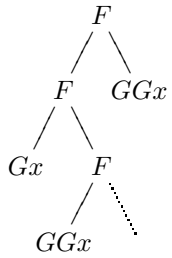
holds, where we write $\frac{1}{3}c$ to mean $\{\, \frac{1}{3}x \mid x \in c \,\}$. The Cantor space arises as the unique solution of a formal equation in the completely iterative algebra formed by all non-empty closed subspaces of $[0, 1]$ metrized with the Hausdorff metric. As our first important result on completely iterative algebras we prove that a final coalgebra for $H(\_) + X$ is precisely the same as a free completely iterative algebra on X. This characterization of final coalgebras as free algebras of some sort is a new and important result of our work. This result opens the door to our subsequent treatment of algebraic semantics using coalgebraic methods. But before that, following Evelyn Nelson's work, we show that free completely iterative algebras yield free completely iterative theories. Similarly, for every *finitary* endofunctor H of Set, i. e., H is determined by its action on finite sets, we formulate the notion of iterative algebra. Then we prove that every set X generates a free iterative algebra RX for H, and we show how to construct every RX from finite coalgebras. As a result of this construction it follows that the initial iterative algebra $R\emptyset$ gives a semantics of all states of finite systems described as coalgebras. For example, when sequential automata are considered as coalgebras, then $R\emptyset$ is the set of all regular languages, i. e., those languages accepted by finite automata. More generally, our construction yields for every polynomial endofunctor of Set the algebras of rational trees as expected. We consider this construction as another one of our main achievements. It is the key to almost all of the

subsequent results on iterativity we present here and in other work. For example, we show that as in the classical setting free iterative algebras yield free iterative theories.

In the second part of this thesis we turn our attention towards applications for semantics of our previously developed theory. We show that our work provides a category-theoretic semantics of so-called recursive program schemes. Classically, a recursive program scheme (RPS) is a system of formal equations such as

$$\begin{array}{lcl} \varphi(x) & \approx & F(x, \varphi(Gx)) \\ \psi(x) & \approx & F(\varphi(Gx), GGx) \end{array} \tag{1.7}$$

where the operations φ and ψ are defined recursively from given operations F and G. The theory of such recursive program schemes is the topic of algebraic semantics, see for instance the book of Irène Guessarian [G]. Actually, one has to distinguish between the uninterpreted semantics of a recursive program scheme and the interpreted one. The uninterpreted semantics provides a meaning of recursive program schemes independently of the actual nature of the given operations. That means that the recursive program scheme is regarded as a purely syntactic construct and its classical uninterpreted semantics will give to each newly defined operation symbol a tree over the signature of givens obtained by unfolding the recursive definition. For example, the above RPS (1.7) has as its uninterpreted solution the infinite trees

$\varphi^\dagger(x) =$  $\psi^\dagger(x) =$  (1.8)

Of course, much has been omitted here. For example, we have not explained why unfolding an RPS yields its solution, or even why the trees in (1.8) are a solution of the RPS (1.7). In order to do this one first needs to define *second-order substitution* of Σ-trees, i. e., substitution of trees for operation symbols, see [C]. Then one can formulate that a solution of an RPS is a fixed point with respect to this second-order substitution; much as solutions of systems of the form (1.4) are fixed points with respect to ordinary or first-order substitution of Σ-trees, i. e., substitution of trees for generators.

In the present thesis we shall formulate the notion of a (suitably generalized) recursive program scheme in a category theoretic setting using final coalgebras in lieu of terms or trees. As it turns out the notion of second-order substitution is elegantly expressed in our setting as a direct consequence of the universal property of free completely iterative theories. This easy but crucial observation allows us to formulate what a solution of an RPS is. We then prove that every guarded recursive program scheme (in our sense) has a unique solution. Guardedness here means that the right-hand sides of an RPS have their head symbols in the signature Σ of given operations. This is also called Greibach normal form in the classical setting. Again, our category theoretic result readily yields the classical one as a special case by working with polynomial endofunctors of Set. But our theory encompasses also applications that go beyond what can be done with the classical methods. For example, we are able to solve recursive program schemes which define operations satisfying equations like commutativity by encoding such extra requirements directly into the RPS. In the classical setting such additional requirements have to be treated separately.

In practise one is often more interested in the interpreted semantics of an RPS. This semantics considers an RPS with the givens from a signature Σ together with a suitable Σ-algebra A, whose operations $\sigma_A : A^n \longrightarrow A$, $\sigma \in \Sigma_n$, $n \in \mathbb{N}$, provide an interpretation of all the given function symbols. Here is the standard example in the subject. Let Σ be the signature of given operation symbols with a constant one, a unary symbol pred, a binary symbol $*$ and a ternary one ifzero. The interpretation we have in mind is the set $\mathbb{N}$ of natural numbers where $\mathsf{ifzero}_{\mathbb{N}}(k, n, m)$ returns m if k is 0 and n otherwise, and all other operations are obvious. The signature Φ of the recursively defined operations consists just of one unary symbol f. Consider the recursive program scheme

$$f(n) \approx \mathsf{ifzero}(n, \mathsf{one}, f(\mathsf{pred}(n)) * n)) \,. \tag{1.9}$$

Then (1.9) is a recursive program scheme defining the factorial function. In general an interpreted RPS is supposed to define new operations on the algebra A in a canonical way such that the formal recursive definitions become valid identities in the given algebra. So by "suitable algebra" we mean, of course, one in which recursive program schemes can be given a semantics. For example, for the recursive program

scheme (1.7) we are only interested in those Σ-algebras A, where $\Sigma = \{F, G\}$, in which the program scheme (1.7) has a *solution*, i. e., we can canonically obtain new operations φ_A and ψ_A on A so that the formal equations (1.7) become valid identities. Before we can actually get to an interpreted semantics the question we have to address is:

$$\text{What } \Sigma\text{-algebras are suitable for semantics?} \tag{1.10}$$

As we mentioned already, several answers have been proposed in the literature. In the classical theory one works with complete posets (cpo) in lieu of sets, see [G]. Here algebras have an additional cpo structure making all operations continuous. Another approach works with complete metric spaces. Here we have an additional complete metric making all operations contracting. In both of these approaches one imposes extra structure on the algebra in a way that makes it possible to obtain the semantics of a recursive computation as a join (or limit, respectively) of finite approximations.

Also (completely) iterative algebras are suitable for semantics. However, they are not the unifying concept one would hope for. While avoiding extra structure they do not subsume continuous algebras on cpos which have *least* (but not necessarily unique) solutions of recursive equations.

The iteration theories of Stephen Bloom and Zoltán Ésik [BÉ] were introduced to overcome this problem, and iteration algebras, see [BÉ], Chapter 7, are algebras that admit canonical (but in general, non-unique) solutions of formal systems of equations.

Analyzing all the above types of algebras we find an interesting common feature which make continuous, metrizable, completely iterative and iteration algebras fit for use in semantics of recursive program schemes: these algebras allow for an evaluation of all Σ-trees. More precisely, for every continuous, metrizable or completely iterative algebra A we obtain a canonical map $T_\Sigma A \longrightarrow A$ which provides for any Σ-tree over A its result of evaluation in A. For completely iterative algebras this follows from the freeness of $T_\Sigma A$ as a completely iterative algebra. It is then easy to give semantics to recursive program schemes in A. For example, for (1.7) one can simply take the tree unfolding which yields the infinite trees of (1.8), and then for any argument $x \in A$ evaluate these infinite trees in A.

Actually, we do not need to be able to evaluate all infinite trees: all recursive program schemes unfold to *algebraic trees*, see [C]. And, as we have seen, another important subclass are the algebras $R_\Sigma X$ of rational trees over X. With this in mind, we can restate problem (1.10) more formally:

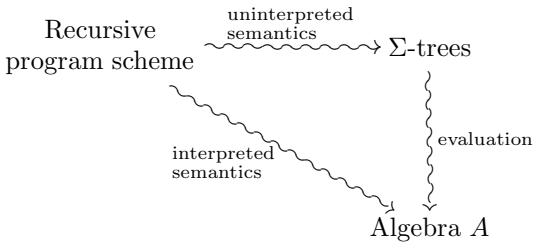
$$\begin{gathered}\text{What } \Sigma\text{-algebras have a suitable evaluation of all trees?} \\ \text{Or all rational trees?}\end{gathered} \tag{1.11}$$

The answer we provide in this thesis is: Σ-algebras carrying a certain additional structure providing canonical solutions of every system of formal equations (1.4). We have chosen to name these algebras (complete) *Elgot algebras* to honour Calvin Elgot whose work has been a great inspiration for us. In contrast to iterative algebras, in an Elgot algebra solutions need not be unique. Instead, there is an operation choosing a solution which is required to satisfy two simple and well-motivated axioms which stem canonically from Elgot's iterative theories. The first of these axioms states that solutions are stable under systematic renaming of the variables of X in system like (1.4). And the second axiom states that simultaneous recursion can be performed sequentially, very similar to what is known as Bekić-Scott law from the theory of least fixed points in complete partial orders. We prove that every free iterative algebra on a set X is also a free Elgot algebra on that set. Moreover, the axioms of Elgot algebras ensure that the evaluation map $R_\Sigma A \longrightarrow A$ obtained by the freeness of $R_\Sigma A$ behaves well with respect to substitution of rational trees for generators. This fact can be expressed in a mathematical precise way in the language of category theory: Elgot algebras for a signature Σ are precisely the Eilenberg-Moore algebras for the monad given by the free iterative theory R_Σ. Hence, one may say that Elgot algebras are precisely those algebras having a canonical evaluation of all rational trees.

Similarly, complete Elgot algebras give canonical solutions to every (not necessarily finite) system of formal equations, where the operation of assigning solutions satisfies the same two simple axioms as above. For a signature Σ we prove that a free complete Elgot algebra on X is precisely the same as a final coalgebra for $H_\Sigma(\_) + X$. And moreover, complete Elgot algebras are precisely the same as the Eilenberg-Moore algebras for the monad given by the free completely iterative theory T_Σ. More elementary, one may say that complete Elgot algebras for a signature Σ are precisely those Σ-algebras with a evaluation map $T_\Sigma A \longrightarrow A$ of all Σ-trees in A. Of course, we introduce (complete) Elgot algebras more generally for endofunctors H, and all our results are proved for this more general category theoretic setting. Basic examples of (complete) Elgot algebras include all continuous algebras, metrizable algebras, and, of course, all (completely) iterative algebras. Using structures for semantics that come with a choice of solutions satisfying certain axioms is not

new. Our two axioms mentioned above are very similar to two of the axioms of iteration theories of Stephen Bloom and Zoltán Ésik [BÉ]. In fact, for a signature Σ with a distinguished constant symbol $\perp$ the rational trees form an iteration theory whose iteration algebras are certain Elgot algebras for Σ satisfying an extra extensionality property.

Once we have established complete Elgot algebras as a suitable class of algebras which can serve as interpretations of givens for an RPS we can present our interpreted RPS semantics. We prove that for every guarded RPS which is given together with a complete Elgot algebra A we can provide a canonical solution in A. Moreover, if A happens to be a completely iterative algebra, this canonical solution is unique. Finally, a fundamental result in the theory of recursive program schemes states that the uninterpreted and interpreted semantics of recursive program schemes are connected via the interpretation of givens in the algebra A:



(1.12)

More detailed, to take first the tree-unfolding of a given recursive program scheme and then compute the resulting Σ-trees in A is the same as to compute the new operations provided by the interpreted semantics of the recursive program scheme. We give a mathematical precise formulation of this consistency between uninterpreted and interpreted RPS solutions, and we show how this can easily be derived in our abstract theory.

We believe that our results in this area generalize and extend the previous work on this topic. Our method for obtaining interpreted solutions easily specializes to the usual denotational semantics using complete partial orders. As a second application we show how to solve recursive program schemes in complete metric spaces. For example, there is a unique contracting function $f : [0,1] \longrightarrow [0,1]$ such that

$$f(x) = \frac{1}{4}\left(x + f\left(\frac{1}{2}\sin x\right)\right). \tag{1.13}$$

Another example: there exists a unique contracting function φ on the set of all non-empty closed subspaces of the Euclidean interval $[0,1]$ satisfying

$$\varphi(x) = \frac{1}{3}\varphi(x) \cup \left(\frac{2}{3} + \frac{1}{3}x\right), \tag{1.14}$$

for any non-empty closed subspace x of $[0,1]$.

Finally, we also provide examples of recursive program schemes and their solutions which cannot be treated within the classical theory: recursive definitions of operations satisfying equations like commutativity.

To sum up, this thesis contributes substantially to a new coalgebraic perspective to recursion which has been developed by researchers in recent years. Our approach of using the theory of coalgebras and the language of category theory to study recursion does not only produce generalizations of well-known results. It also provides new tools and offers new insights into the general mechanisms at work in the semantics of recursion at a concise and conceptual level. Furthermore, we claim that our abstract categorical approach allows to bring several well-known approaches to semantics of recursion under one roof. And, as we have seen, our work is starting to produce new results in semantics.

The Structure of this Text

This summary is structured as follows: in Section 2 we will briefly recall some notions and results from category theory which are perhaps less familiar; in particular we discuss locally finitely presentable categories and *monads*. For the convenience of the reader we also provide a bit of introduction to the theories of algebras and coalgebras for an endofunctor. The material of the papers summarized in this document begins in Section 3. We study first iterative and completely iterative algebras. This section presents the results of

the first parts of our papers [M_1] and [AMV_2], the latter of which is joint work with Jiří Aámek and Jiří Velebil.

Then we start to study the monads of free (completely) iterative algebras and we characterize in Section 4 the Eilenberg-Moore algebras for these monads—they are precisely (complete) Elgot algebras. We argue that Elgot algebras capture important aspects of different types of algebras commonly used in semantics of recursion. In particular, we show that continuous algebras and algebras on complete metric spaces are complete Elgot algebras. These are the results of [AMV_3], which is joint work with Jiří Aámek and Jiří Velebil, again. In Section 5, we show that much more complicated systems of recursive definitions than the ones of Section 3 can be solved uniquely in (completely) iterative algebras. These results are the basis for the work in Section 6 where the monads of free (completely) iterative algebras are shown to be the free (completely) iterative monads generated by an endofunctor. So Sections 5 and 6 present the second parts with the main results of the papers [M_1, AMV_2] and the results of [M_2]. While these results are mathematically quite pleasing they also are an important step towards our semantics of (generalized) recursive program schemes in Section 7, which is the topic of the joint paper with Larry Moss [MM]. In fact, the universal property of free completely iterative monads yields as a special case the so-called second-order substitution, which will allow us to define what a solution of a recursive program scheme in our abstract setting is. We will then present the uninterpreted as well as an interpreted semantics of recursive program schemes. Our main results in Section 7 are that every guarded recursive program scheme can be given a unique uninterpreted solution. Furthermore, using complete Elgot algebras as interpretations of givens we provide a canonical interpreted semantics, and we show that interpreted and uninterpreted solutions are consistent with each other. Finally, we illustrate our results with several applications.

We conclude this summary with some discussion of presently ongoing and directions for future research in the last Section 8.

Related Work

As we have mentioned already it was the idea of Calvin Elgot to study the semantics of recursive definitions at a purely algebraic level. He introduced iterative algebraic theories in [E], and later he proved with his coauthors that free iterative theories exist [BE] and are given by rational trees [EBT], see also the work of Susanna Ginali [Gi_1, Gi_2] for the first proof of this fact. Similarly, free completely iterative theories exist and are given by all infinite trees.

Iterative algebras were introduced and studied by Jerzy Tiuryn [T] and Evelyn Nelson [N] with the aim of providing a more easy approach to Elgot's iterative theories. Nelson proved that rational trees yield free iterative algebras and that those give a free iterative theory.

The first categorical accounts of infinite trees as monads of final coalgebras appear independently and almost at the same time in the work of Larry Moss [Mo_1], of Neil Ghani, Christoph Lüth, Federico De Marchi and John Power [$GLMP_1$, $GLMP_2$], and of Peter Aczel, Jiří Adámek and Jiří Velebil [AAV]. Furthermore, in [Mo_1] and [AAV] it is proved that those monads are completely iterative. And in [AAMV] we established the universal property of the free completely iterative monads. The categorical treatment of rational trees and iterative theories started with our work [AMV_1]. We proved there that every endofunctor of Set admits a free iterative monad and we provided a coalgebraic construction of this monad. In [GLM_1] the authors gave a general construction based on our ideas that allows to obtain (monads of) other syntactic objects than rational trees with a coalgebraic construction.

The axioms of (complete) Elgot algebras as presented in Section 4 below were inspired by the axioms of iteration theories of Stephen Bloom and Zoltán Ésik [BÉ]. Other approaches using solutions and axioms are traced monoidal categories of André Joyal, Ross Street and Dominic Verity [JSV], see the traced cartesian categories in [Ha] for the connection, or fixed-point theories for domains, see the work of Samuel Eilenberg [Ei] or Gordon Plotkin [P], etc.

The classical theory of recursive program schemes is presented by Irène Guessarian [G]. There one finds results on uninterpreted solutions of program schemes and interpreted ones in continuous algebras.

Basic properties of infinite trees are presented by Bruno Courcelle in [C]. Our work [AAMV, M_1] gives a categorical description of second-order substitution of infinite trees which we use in order to formulate solutions of recursive program schemes abstractly.

Neil Ghani, Christoph Lüth and Federico De Marchi [GLM_2] obtained a general solution theorem with the aim of providing a categorical treatment of uninterpreted program scheme solutions. Part of our proof for the solution theorem for uninterpreted schemes is inspired by their proof of the same fact. However, the connection to (generalized) second-order substitution as presented in [AAMV, M_1] is new in our work.

Complete metric spaces as a basis for the semantics of recursive program schemes have been studied by André Arnold and Maurice Nivat, see e. g. [AN]. Stephen Bloom [Bl] studied interpreted solutions of recursive program schemes in so-called contraction theories. The semantics of recursively defined data types as fixed points of functors on the category of complete metric spaces has been investigated by Pierre America and Jan Rutten [ARu] and by Jiří Adámek and Jan Reitermann [ARe]. We build on this with our treatment of self-similar objects. These have also recently been studied in a categorical framework by Tom Leinster, see [Le_1, Le_2, Le_3].

2 Preliminaries

> We will need to use some very simple notions of category theory,
> an esoteric subject noted for its difficulty and irrelevance.
> Moore and Seiberg [MS]

In this section we collect most of the basic notions and standard results used frequently throughout this summary. We assume that the reader is acquainted with some basics of category theory like categories, functors, natural transformations, limits, colimits, etc., see any standard text on category theory, e.g. the excellent textbook [ML] by Saunders MacLane. Everything else necessary to understand the work in the subsequent sections shall be mentioned here. This section is not thought as a general introduction to any of the theories treated in the following subsections. All we want to do here is to provide a little bit of intuition for the basic material used in our subsequent work, set up notation and recall the usual results. We will mostly not give proofs but provide references where arguments are not obvious. Readers familiar with category theory, algebras and coalgebras should skip this section and start reading Section 3 immediately.

2.1 Locally Finitely Presentable Categories

For some of our work it will be necessary to work in a category in which it makes sense to speak about a "finite object" in the same way as one can talk about finite sets in the category Set of (small) sets and functions. Moreover, we shall be interested in working with finitary endofunctors, i. e., those functors which are determined by their action on "finite objects". Such an environment is provided by locally finitely presentable categories in the sense of Peter Gabriel and Friedrich Ulmer [GU]. Before we give the definition let us recall the notions of filtered diagram and finitely presentable object.

Definition 2.1.

(i) A category $\mathcal{D}$ is called *filtered* if each finite subcategory has a cocone in $\mathcal{D}$, or more precisely

  (a) $\mathcal{D}$ is non-empty,

  (b) for each pair D_1, D_2 of objects of $\mathcal{D}$ there exists a cospan in $\mathcal{D}$, i.e, an object D and two morphisms $f_i : D_i \longrightarrow D$, $i = 1, 2$, and

  (c) for each pair of parallel morphisms $f, g : D \longrightarrow D'$ in $\mathcal{D}$ there exists a coequating morphism, i. e., a morphism $h : D' \longrightarrow D''$ in $\mathcal{D}$ such that $h \cdot f = h \cdot g$.

(ii) A *filtered diagram* is a diagram $D : \mathcal{D} \longrightarrow \mathcal{A}$ whose scheme $\mathcal{D}$ is a filtered category, and a *filtered colimit* is a colimit of a filtered diagram.

(iii) A functor $F : \mathcal{A} \longrightarrow \mathcal{B}$ is called *finitary* provided that it preserves filtered colimits.

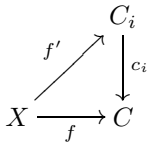
(iv) An object X of a category $\mathcal{A}$ is called *finitely presentable* if its covariant hom-functor $\mathcal{A}(X, -)$ is finitary.

Notation 2.2. For two categories $\mathcal{A}$ and $\mathcal{B}$ we shall denote by

$$[\mathcal{A}, \mathcal{B}] \qquad \text{and} \qquad \mathsf{Fin}[\mathcal{A}, \mathcal{B}]$$

the category of all endofunctors and of finitary ones, respectively.

Remark 2.3. The following more explicit characterization of finite presentability of an object X will be used frequently in our work: X is finitely presentable if and only if for any filtered colimit C and any morphism $f : X \longrightarrow C$, there exists an essentially unique factorization through an injection of the colimit. More detailed, there exists an injection $c_i : C_i \longrightarrow C$ of the colimit and a morphism $f' : X \longrightarrow C_i$ such that the triangle



commutes, and furthermore, whenever two factorizations f' and f'' of f through any colimit injection $c_j : C_j \longrightarrow C$ satisfy $c_j \cdot f' = c_j \cdot f''$, then f' and f'' can be coequated in the diagram, i. e., there exists a morphism $c_{jk} : C_j \longrightarrow C_k$ in the diagram such that $c_{jk} \cdot f' = c_{jk} \cdot f''$.

Next, we list some examples of finitely presentable objects. More examples can be found in [AR].

Examples 2.4.

(i) A set X is finitely presentable in Set if and only if it is a finite set.

(ii) Similarly, in Pos the category of partially ordered sets and order preserving maps an object is finitely presentable if and only if it is a finite poset.

(iii) A group is finitely presentable in the category Grp of groups and their homomorphisms if and only if it can be presented by finitely many generators and finitely many equations. More generally, an object in a variety of finitary (many-sorted) algebras is finitely presentable if and only if it can be presented by finitely many generators and equations.

(iv) Consider the category CPO of complete partial orders and continuous functions between them. So objects of CPO are partially ordered sets (not necessarily with a least element) in which every ascending chain has a join, and morphisms are functions preserving such joins. In CPO, only the empty cpo is finitely presentable.

The following result about finitely presentable objects will often be used, see [AR], Proposition 1.3.

Proposition 2.5. *A finite colimit of finitely presentable objects is finitely presentable.*

Definition 2.6. A category $\mathcal{A}$ is called *locally finitely presentable* (or, *lfp*, for short) if

(i) it is cocomplete,

(ii) it has (up to isomorphism) only a set of finitely presentable objects, and

(iii) every object of $\mathcal{A}$ is a filtered colimit of finitely presentable objects.

Examples 2.7.

(i) Set is lfp since every set is a directed union of its finite subsets, and the collection of all finite sets is up to isomorphism a (countable) set.

(ii) Pos, Grp, and every variety of finitary (many-sorted) algebras are lfp.

(iii) CPO is not lfp.

(iv) A poset considered as a category is lfp if and only if it is a complete algebraic lattice.

Next, we list properties of lfp categories used in this summary. We omit all proofs as they can be found in [AR].

Notation 2.8. For any lfp category $\mathcal{A}$ we denote by $\mathcal{A}_{\mathrm{fp}}$ the full subcategory of $\mathcal{A}$ given by (a representing set of) finitely presentable objects.

Proposition 2.9. *Every object A of an lfp category $\mathcal{A}$ is a colimit of the canonical filtered diagram*

$$D_A : \mathcal{A}_{\mathrm{fp}}/A \longrightarrow \mathcal{A}, \qquad (X \longrightarrow A) \longmapsto X\,,$$

of finitely presentable objects, i. e., $\mathcal{A}_{\mathrm{fp}}$ is a dense subcategory.

Theorem 2.10. *Any lfp category is complete and well-powered as well as cocomplete and cowell-powered.*

Theorem 2.11. (Adjoint Functor Theorem)
A functor between lfp categories is a right adjoint if and only if it preserves limits and filtered colimits.

Proposition 2.12. *Let $\mathcal{A}$ and $\mathcal{B}$ be lfp categories.*

(i) The categories $[\mathcal{A}_{\mathrm{fp}}, \mathcal{B}]$ and $\mathsf{Fin}[\mathcal{A}, \mathcal{B}]$ are equivalent.

(ii) For two lfp categories $\mathcal{A}$ and $\mathcal{B}$ the category $\mathsf{Fin}[\mathcal{A}, \mathcal{B}]$ is lfp.

In fact, the first item follows from the fact that $\mathcal{A}$ is a free cocompletion under filtered colimits of $\mathcal{A}_{\mathrm{fp}}$.

Remark 2.13. Analogously to lfp categories there exists the concept of locally λ-presentable category, for λ a regular cardinal, see [AR]. A category $\mathcal{A}$ is locally λ-presentable if it is cocomplete and has (up to isomorphism) a set of λ-presentable objects such that any object of $\mathcal{A}$ is a λ-filtered colimit of objects from that set.

Here the notions of λ-filtered colimit and λ-presentability are defined as expected: a category $\mathcal{D}$ is called λ-filtered if every subcategory with less than λ morphisms has a cocone in $\mathcal{D}$, colimits of diagrams whose domain is λ-filtered are called λ-filtered colimits, etc. Notice that a functor F is called *λ-accessible* if it preserves λ-filtered colimits, and F is called *accessible* if it is λ-accessible for some regular cardinal λ. Observe that Set is locally λ-presentable for every regular cardinal λ. Recall that an endofunctor $H : \mathsf{Set} \longrightarrow \mathsf{Set}$ is called *bounded* if there exists a cardinal number λ such that for every set X any element $x \in HX$ lies in the image of Hm for some set $m : Y \hookrightarrow X$ of X of cardinality less that λ. Accessible endofunctors of Set are precisely the bounded ones, see [AT], Proposition III.4.3.

In this text we will not work with arbitrary locally λ-presentable categories. But we shall often mention and use accessible endofunctors (of Set).

2.2 Algebras

In this subsection we shall recall some basics on algebras for an endofunctor. Their purpose within our work is to provide a categorical framework for the notion of a set equipped with operations. We will provide here enough intuition to make this precise. In Sections 3 and 4 we will study certain algebras suitable for semantics of recursive programs, i.e., algebras in which it is possible to define new operations from given ones by means of a recursive specification.

Another view on algebras for an endofunctor is as implementations of abstract data types. More precisely, an *initial* algebra provides a canonical semantics of an abstract data type whose constructors are described by means of a suitable endofunctor of a category.

Definition 2.14. Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be an endofunctor of a category $\mathcal{A}$. By an *H-algebra* we mean a pair (A, a) where A is an object of $\mathcal{A}$, the *carrier* of the algebra, and

$$a : HA \longrightarrow A$$

is a morphism of $\mathcal{A}$, the *structure* of the algebra.

An *(H-algebra) homomorphism* from an algebra (A, a) to an algebra (B, b) is a morphism $h : A \longrightarrow B$ of $\mathcal{A}$ preserving the algebraic structure, i. e., such that the square

$$\begin{array}{ccc} HA & \xrightarrow{a} & A \\ {\scriptstyle Hh}\downarrow & & \downarrow{\scriptstyle h} \\ HB & \xrightarrow[b]{} & B \end{array} \tag{2.15}$$

commutes.

Notation 2.15. Observe that the identity on a carrier of an algebra is a homomorphism and that homomorphisms compose. Thus, H-algebras and their homomorphisms organize themselves in a category which we will denote by

$$\mathsf{Alg}\, H\,.$$

Remark 2.16. An important concept is that of an initial algebra. An H-algebra (I, i) is initial if for every H-algebra (A, a) there exists precisely one homomorphism from I to A, i. e., (I, i) is an initial object of $\mathsf{Alg}\, H$. Notice that, being a colimit, an initial algebra is uniquely determined up to isomorphism.

Examples 2.17.

(i) Consider the endofunctor of Set given by $HX = X + 1$. Algebras for H are unary algebras with a constant, and homomorphisms are maps preserving the constant and the unary operation. An initial algebra is the algebra $\mathbb{N}$ of natural numbers together with the constant 0 and the successor function. Notice that the elements of the initial algebra can be thought of as being build up inductively starting with the constant, then applying formally the unary operation to form new elements. In fact, the initiality of $(\mathbb{N}, 0, \mathsf{succ})$ can be understood as induction principle: to define a function f from $\mathbb{N}$ to a

set A it suffices to specify its value at 0, i. e., an element $a = f(0)$, and a function $\alpha : A \longrightarrow A$ which will tell us that $f(n+1) = \alpha(a')$ if we already have defined $f(n) = a'$ for some natural number n. Indeed, (A, a, α) yields an H-algebra. Thus, by the initiality there exists a function $f : \mathbb{N} \longrightarrow A$ and the fact that it is an algebra homomorphism is equivalent to saying that $f(0) = a$ and $f(n+1) = \alpha(f(n))$, for any natural number $n \geq 1$, hold. The uniqueness of f yields an induction proof principle: to prove that two functions $f, g : \mathbb{N} \longrightarrow A$ are equal it suffices to establish that both f and g are algebra homomorphisms: then $f(0) = g(0)$ and whenever $f(n) = g(n)$, then also $f(n+1) = \alpha(f(n)) = \alpha(g(n)) = g(n+1)$. Thus, $f = g$ as desired.

(ii) Binary algebras and their homomorphisms are precisely the algebras for the set endofunctor given by $HX = X \times X$. In this case the initial algebra is carried by the empty set.

(iii) Lists. Consider the functor on Set given by $HX = 1 + S \times X$ for a fixed set S. The H-algebras are algebras A with a constant and unary operations α_s for each $s \in S$—the latter can, of course, be presented as one map $\alpha : S \times A \longrightarrow A$. Homomorphisms are, again, the structure preserving maps. An initial algebra is given by the set S^* of finite lists over S with the constant given by the empty list and with the unary operations given by concatenation:

$$\begin{array}{rcl} \mathsf{cons} : S \times S^* & \longrightarrow & S^* \\ (s, \langle s_1, \ldots, s_n \rangle) & \longmapsto & \langle s, s_1, \ldots, s_n \rangle \,. \end{array}$$

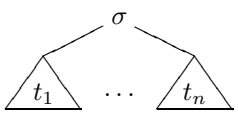
As before the initiality gives rise to an induction principle, i. e., the initiality is precisely the fact that functions out of S^* can be uniquely defined by structural induction. We leave it to the reader to work out the details.

(iv) Trees.[1] For two sets of labels L and M consider the functor $H : \mathsf{Set} \longrightarrow \mathsf{Set}$ defined by $HX = L \times X \times X + M$. Algebras of H have a binary operation for each element $l \in L$ and constants for each $m \in M$, and homomorphisms preserve these operations and constants. An initial H-algebra is the algebra T of all binary trees with inner nodes labelled by elements of L and leaves labelled by elements of M. The algebra structure $L \times T \times T + M \longrightarrow T$ assigns to an element of M the single-node tree labelled by that element, and to a triple (l, t_1, t_2) the binary tree obtained by joining the trees t_1 and t_2 with a common root node which is labelled by l.

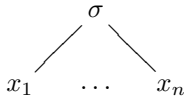
(v) The previous examples (i) to (iv) are subsumed by general algebras for a signature. Let Σ be a signature (or, ranked alphabet), i. e., Σ gives for each natural number n a set Σ_n of operation symbols of *arity* n. We form the polynomial functor $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$ associated to Σ by defining

$$H_\Sigma X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X^2 + \cdots \tag{2.16}$$

on objects and similarly on morphisms. It is not difficult to see that $\mathsf{Alg}\, H_\Sigma$ is the category of Σ-algebras, i. e., general algebras for the signature Σ, and their homomorphisms. An initial algebra is the algebra F_Σ of all finite Σ-trees (or, Σ-terms), i. e., finite ordered trees where each node with n children is labelled by an n-ary operation symbol. The algebra structure map $\varphi_\Sigma : H_\Sigma F_\Sigma \longrightarrow F_\Sigma$ is given by tree tupling, more precisely, on the n-th coproduct component of $H_\Sigma F_\Sigma$, φ_Σ assigns to a tuple $(\sigma, t_1, \ldots, t_n)$ where σ is an n-ary operation symbol and $t_1, \ldots, t_n$ are finite Σ-trees from F_Σ the tree



Finally, notice that the elements of $H_\Sigma X$ for a set (of variables) X can be understood as *flat trees*, i. e., trees whose root node is labelled by an n-ary operation symbol and where this root has n-children which are leaves labelled by elements from X:



We will sometimes use the notation $(\sigma, \vec{x})$ for elements of $H_\Sigma X$.

[1] Trees are throughout be understood as rooted ordered ones up to isomorphism.

(vi) Algebras with operations that satisfy certain equations can sometimes be expressed as algebras for an endofunctor. For example, the algebras with a binary commutative operation are the algebras for the functor $\mathcal{P}_2$ on Set assigning to a set X the set of unordered pairs

$$\mathcal{P}_2 : X \longmapsto \{\, \{\, x, y \,\} \mid x, y \in X \,\}\,.$$

Of course, H-algebra homomorphisms are precisely the morphisms preserving the binary operation. The initial algebra is carried by the empty set.

(vii) More generally, algebras for finitary endofunctors of Set can be understood as algebras for a signature satisfying equations. Recall that a functor $H : \mathsf{Set} \longrightarrow \mathsf{Set}$ is finitary (i. e., it preserves filtered colimits, see Definition 2.1) if and only if it is a quotient of some polynomial functor H_Σ, see [AT], III.4.3. The latter means that we have a natural transformation $\varepsilon : H_\Sigma \longrightarrow H$ with epimorphic components ε_X, which are fully described by their kernel equivalence whose pairs can be presented in the form of so-called *basic equations*

$$\sigma(x_1, \ldots, x_n) = \varrho(y_1, \ldots, y_m)$$

for $\sigma \in \Sigma_n$, $\varrho \in \Sigma_m$ and $(\sigma, \vec{x}), (\varrho, \vec{y}) \in H_\Sigma X$ for some set X including all x_i and y_j. It is not difficult to prove that the algebras for H are precisely the Σ-algebras satisfying the basic equations given by ε. The initial algebra I of H is given by the quotient $I_\Sigma/\!\sim$ where $\sim$ is the smallest congruence on I_Σ such that $I_\Sigma/\!\sim$ is an H-algebra. More explicitly, for finite Σ-trees s and t we have $s \sim t$ if and only if t can be obtained from s by finitely many applications of the basic equations describing ε. For example, the functor H which assigns to a set X the set $\{\, \{\, x, y \,\} \mid x, y \in X \,\} + 1$ of unordered pairs of X plus an extra element is a quotient of $H_\Sigma X = X \times X + 1$ expressing one binary operation b and a constant c, where ε_X is presented by commutativity of b; i. e., by the basic equation $b(x, y) = b(y, x)$. And I is the algebra of all finite unordered binary trees.

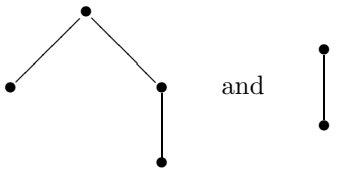
(viii) It follows from (vii) above that a finitary variety of Σ-algebras for a signature forms a category of algebras for a finitary endofunctor of Set if and only if the variety can be defined by basic equations. We already saw in (vi) above algebras with a commutative binary operation as algebras for the endofunctor $\mathcal{P}_2$. However, the category of semigroups (i. e., algebras with one associative binary operation) does not form a category of algebras for a finitary set endofunctor.

(ix) Consider the finite power set functor $\mathcal{P}_{\mathrm{fin}}$ assigning to a set X the set of its finite subsets. We do not give an explicit description of $\mathsf{Alg}\,\mathcal{P}_{\mathrm{fin}}$, but we describe the initial algebra. The functor $\mathcal{P}_{\mathrm{fin}}$ is given as a quotient $\varepsilon : H_\Sigma \longrightarrow \mathcal{P}_{\mathrm{fin}}$ where Σ has for each natural number n a unique n-ary operation symbol, i. e., $H_\Sigma X = \coprod_n X^n$, and $\varepsilon_X(x_1, \ldots, x_n) = \{\, x_1, \ldots, x_n \,\}$ for every n-tuple of elements of X. Thus, the initial $\mathcal{P}_{\mathrm{fin}}$-algebra is given by a quotient of the initial H_Σ-algebra modulo basic equations as in (vii) above. We give alternative descriptions here.

An initial $\mathcal{P}_{\mathrm{fin}}$-algebra I is carried by the set of all finite constructive sets. Then $I = \mathcal{P}_{\mathrm{fin}} I$ so that the structure of the initial algebra is the identity map. Before we give another (isomorphic) description of I we need to recall the following notions. A *bisimulation* between two unordered trees s and t is a relation $\sim$ between their sets of nodes with the following property: if $n \sim m$ are related nodes of s and t, respectively, then for each child n' of n there exists a child m' of m with $n' \sim m'$, and vice versa, for each child m' of m there exists a child n' of n with $n' \sim m'$. The trees s and t are called *bisimilar* if there exists a bisimulation between them relating their root nodes. For example, the two trees

(2.17)

are bisimilar, whereas



are not. An unordered tree t is called *strongly extensional* if for each node two distinct subtrees rooted at children of that node are never bisimilar. Notice that, the greatest bisimulation on a tree t is always

an equivalence relation and the quotient of t modulo this equivalence is a strongly extensional tree. For example, consider the two trees in (2.17) above. The left-hand one is not strongly extensional, but the right-hand one is and in fact, it is the stronly extensional quotient of the left-hand tree.

Now coming back to the initial algebra I of $\mathcal{P}_{\mathrm{fin}}$, one can prove that I consists of all unordered finite strongly extensional trees.

Remark 2.18. Despite the fact, that all our previously presented examples are algebras for set endofunctors it is useful to work in an abstract category $\mathcal{A}$. Firstly, in our proofs and constructions it will be clearer what concepts and structure really have to be employed to make things work. Secondly, there may later arise applications that call for the use of other ground categories than Set.

In fact, we shall later see applications in semantics of computation where we deal with algebras (e. g. for a signature) having extra structure. For example, continuous algebras are carried by a complete partial order rather that just a set. Likewise, one may be interested to work with algebras on complete metric spaces. Those algebras can conveniently be treated by considering algebras for a functor on the category of complete partial orders and continuous maps or the category of complete metric spaces and non-expanding maps, respectively.

Next we give some basic results on algebras, which we shall frequently use throughout our work. First of all, constructions like products of algebras etc. can be performed on the level of the underlying category $\mathcal{A}$. Recall that a functor $F : \mathcal{A} \longrightarrow \mathcal{B}$ is said to *create* limits if for any diagram $D : \mathcal{D} \longrightarrow \mathcal{A}$ and for every limiting cone $(L, (\ell_i)_{i\in\mathcal{D}})$ of $F \cdot D$ there exists a unique cone $(A, (a_i)_{i\in\mathcal{D}})$ with $FA = L$ and $Fa_i = \ell_i$, for all $i \in \mathcal{D}$, and this cone is a limit of D. For the forgetful functor $U : \mathsf{Alg}\, H \longrightarrow \mathcal{A}$ creation of limits means that for each diagram D of algebras and homomorphisms the limit L of the diagram $U \cdot D$ in $\mathcal{A}$ can uniquely be equipped with an algebra structure so that all limit projection become homomorphisms and the resulting cone is a limit in $\mathsf{Alg}\, H$ of D.

Theorem 2.19. *The forgetful functor $U : \mathsf{Alg}\, H \longrightarrow \mathcal{A}$ creates all limits and all colimits that H preserves.*

In general, colimits of algebras are not formed on the level of $\mathcal{A}$, e. g., coproducts of binary algebras are not carried by the disjoint union of the carriers. However, there are results on the cocompleteness of $\mathsf{Alg}\, H$, see [Li$_2$, AK$_1$] or [AK$_2$].

The next result states that an initial algebra is the least fixed point of H, or the least solution of the recursive domain equation

$$HX \simeq X\,.$$

Lemma 2.20. (Lambek [L])
If (I, i) is an initial H-algebra, then its structure is an isomorphism.

Proof. Clearly, (HI, Hi) is an H-algebra. Thus there exists a unique homomorphism j from (I, i) to (HI, Hi). Since $i : HI \longrightarrow I$ is a homomorphism, we obtain the commutative diagram

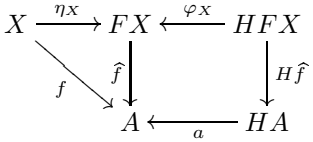
$$\begin{array}{ccc} HI & \xrightarrow{\;i\;} & I \\ {\scriptstyle Hj}\downarrow & & \downarrow{\scriptstyle j} \\ HHI & \xrightarrow{\;Hi\;} & HI \\ {\scriptstyle Hi}\downarrow & & \downarrow{\scriptstyle i} \\ HI & \xrightarrow[\;i\;]{} & I \end{array}$$

Thus, we have $i \cdot j = id$ by the initiality, because id is the only homomorphism from the initial algebra to itself. The commutativity of the upper square now gives $j \cdot i = Hi \cdot Hj = id$. □

Another very important concept is that of a free algebra on a given object of generators.

Definition 2.21. Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be an endofunctor, and let X be an object of $\mathcal{A}$. An H-algebra (FX, φ_X) together with a morphism $\eta_X : X \longrightarrow FX$ is called a *free H-algebra on X* if for every H-algebra (A, a) and every morphism $f : X \longrightarrow A$ there exists a unique homomorphic extension $\widehat{f} : FX \longrightarrow A$, i. e., a unique

H-algebra homomorphism $\widehat{f} : (FX, \varphi_X) \longrightarrow (A, a)$ with $\widehat{f} \cdot \eta_X = f$:



Remark 2.22.

(i) In a category $\mathcal{A}$ with finite coproducts, it is easy to show that for every object X an initial algebra for the functor $H(\_) + X$ is precisely a free H-algebra on X. More detailed, if FX is a free H-algebra on X, then its algebraic structure $\varphi_X : HFX \longrightarrow FX$ and the universal arrow $\eta_X : X \longrightarrow FX$ yield the structure of the initial algebra for $H(\_) + X$, and conversely.

Observe that the free H-algebra $F0$ on the initial object of $\mathcal{A}$ is precisely the initial H-algebra.

(ii) Functors H having free algebras on every object X are called *varietors* in [AT]. For a varietor the assignment $X \longmapsto FX$ of a set to a free algebra on that set yields a left-adjoint to the forgetful functor $U : \mathsf{Alg}\, H \longrightarrow \mathcal{A}$. All functors of Example 2.17 are varietors of Set. However, not every set functor is a varietor. The (unbounded) power set functor $\mathcal{P} : \mathsf{Set} \longrightarrow \mathsf{Set}$ does not have an initial algebra, whence no free algebras. In fact, there can be no fixed point of $\mathcal{P}$ due to the famous Cantor Theorem.

(iii) Varietors do not enjoy nice closure properties. For example, for $\mathcal{A} = \mathsf{Set}$ varietors need not compose, nor need a coproduct of two varietors be a varietor, see [AT], IV.4.4. Therefore, one often works with a more nicely behaved subclass of varietors, for example finitary (or, more generally, accessible) functors of Set, see Theorem 2.25 below.

Examples 2.23.

(i) For a polynomial endofunctor $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$ induced by a signature a free algebra on a set X is carried by the set $F_\Sigma X$ of all finite Σ-trees over X, i. e., Σ-trees where leaves can be labelled either by a generator from X or by a constant symbol from Σ. We leave it to the reader to work out the special cases for the functors of Examples 2.17(i)–(iv). Observe that the freeness of $F_\Sigma X$ specializes to substitution of terms for variables. In fact, let X and Y be sets of variables. Then any map $s : X \longrightarrow F_\Sigma Y$ gives a substitution of variables of X by finite Σ-trees over Y. The unique induced homomorphism $\widehat{s} : F_\Sigma X \longrightarrow F_\Sigma Y$ performs for any finite Σ-tree t in $F_\Sigma X$ the substitution s, i. e., every leaf labelled by $x \in X$ is replaced by $s(x) \in F_\Sigma Y$.

Notice also that for any Σ-algebra A we obtain a canonical homomorphism $\alpha : F_\Sigma A \longrightarrow A$ extending the identity on A. It provides evaluations of all finite Σ-trees over A in the algebra A. In fact, one easily verifies that α is obtained by canonically extending the computation of all flat Σ-trees over A which is provided by the algebraic structure $a : H_\Sigma A \longrightarrow A$.

(ii) For an arbitrary finitary functor H of Set which comes as a quotient $\varepsilon : H_\Sigma \longrightarrow H$ we have described an initial algebra in Example 2.17(vii). Analogously, a free algebra FX on a set X is a quotient $F_\Sigma X/{\sim_X}$ of the free H_Σ-algebra on X. In fact, notice that we obtain a natural transformation $\varepsilon' = \varepsilon + id : H_\Sigma(\_) + X \longrightarrow H(\_) + X$ exhibiting $H(\_) + X$ as a quotient of the polynomial functor $H_\Sigma(\_) + X$. Observe that the basic equations for ε' are precisely the same as those for ε, and recall Remark 2.22(i) to obtain FX as the quotient of $F_\Sigma X$ modulo basic equations; more detailed, $s \sim_X t$ holds for Σ-trees s and t over X if s is obtained from t by application of finitely many basic equations provided by ε.

(iii) A free $\mathcal{P}_2$-algebra on a set X where $\mathcal{P}_2$ is the unordered pair functor from Example 2.17(v) is carried by the set of binary unordered trees with leaves labelled in X. The algebra structure is given by tree pairing.

(iv) A free $\mathcal{P}_{\mathrm{fin}}$-algebra on a set X is carried by the set of finitely branching strongly extensional finite trees with leaves partially labelled in X.

A free H-algebra on an object X can in many categories be inductively constructed. We describe this first for the case of an endofunctor H on Set. For a set X we define by transfinite induction the following chain indexed by ordinal numbers:

Initial step: $H_0X = X$

$h_{0,1} \equiv H_0X = X \xrightarrow{\mathsf{inr}} HX + X = H_1X$

Isolated step: $H_{i+1}X = HH_iX + X$

$h_{i+1,j+1} \equiv H_{i+1}X = HH_iX + X \xrightarrow{Hh_{i,j}+X} HH_j + X = H_{j+1}X$

Limit step: $H_jX = \operatorname{colim}_{i<j} H_iX$ with injections $h_{i,j} : H_iX \longrightarrow H_jX$, $i < j$,

the connecting map $h_{j,j+1}$ is uniquely determined by the commutativity of the squares

$$\begin{array}{ccc} H_iX & \xrightarrow{\quad h_{i,j} \quad} & H_jX \\ {\scriptstyle h_{i,i+1}}\downarrow & & \vdots\,{\scriptstyle h_{j,j+1}} \\ H_{i+1}X = HH_iX + X & \xrightarrow[Hh_{i,j}+X]{} & HH_jX + X = H_{j+1}X \end{array} \qquad i < j\,.$$

The above chain is said to *converge* if $h_{i,i+1}$ is an isomorphism for some ordinal number i.

Theorem 2.24. ([AT], Theorem IV.4.2)
Let H be an endofunctor of Set. Then H has a free algebra on X if and only if the above chain (H_iX) converges. Furthermore, if $h_{i,i+1}$ is invertible for some i, then $(H_iX, h_{i,i+1}^{-1})$ is a free H-algebra on X.

This result does not indicate after how many steps the free algebra (if it exists) is constructed. However, the following theorem gives a bound for endofunctors H that preserve colimits of λ-chains, i. .e, diagrams $\lambda \longrightarrow \mathsf{Set}$ for an infinite ordinal λ.

Theorem 2.25. (Adámek [A_1])
Let H be an endofunctor that preserves colimits of λ-chains. Then the free algebra chain converges after λ steps, and the pair $(H_\lambda X, h_{\lambda,\lambda+1}^{-1})$ is a free H-algebra on X.

This last result is not specific for Set. In fact, its proof works in any category $\mathcal{A}$ in which finite coproducts and colimits of α-chains, $\alpha \leq \lambda$, exist. In particular, if $\mathcal{A}$ is an lfp category and H is a finitary functor, then H preserves colimits of ω-chains, thus the free algebra construction stops after ω steps. Finally, notice that in the light of Remark 2.22(i) the above construction easily specializes to a construction of an initial algebra. In fact, just take for X the empty set (or, the initial object of $\mathcal{A}$).

2.3 Coalgebras

In this subsection we recall coalgebras for an endofunctor. Whereas algebras can be thought of as sets of data elements equipped with operations that "construct" new data elements (see the examples of natural numbers, lists and trees as initial algebras) coalgebras have a strong flavour of dynamic systems where a set of states is equipped with "observations" that allow to access certain information about these states albeit not the states themselves. The type of the system, i. e., the type of the possible observations of the systems, is described by the endofunctor under consideration. Most prominently, sequential automata and labelled transition systems can be considered as coalgebras, see Examples 2.29 below. Another perspective on coalgebras is the one viewing them as infinitary data structures like streams or infinite trees. Coalgebraic principles are then useful to define operations on such data structures as well as proving properties about such operations. Again, it is not our aim to provide a general introduction to the theory of coalgebras but merely collect those parts which will be essential to our work. For a general introduction to the field of coalgebra the reader may consult any of the texts [JR, R_2, Gu, A_2].

Formally, coalgebras are dual as objects to algebras in the sense that the structure is turned around.

Definition 2.26. Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be an endofunctor of a category $\mathcal{A}$. An H-coalgebra is a pair (A, a) consisting of a *carrier* A, an object of $\mathcal{A}$, and the *structure* $a : A \longrightarrow HA$. A coalgebra homomorphism h

from (A, a) to (B, b) is a morphism $h : A \longrightarrow B$ such that the square

$$\begin{array}{ccc} A & \xrightarrow{a} & HA \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh} \\ B & \xrightarrow[b]{} & HB \end{array} \tag{2.18}$$

commutes.

Remark 2.27.

(i) Coalgebras for a functor H and their homomorphisms form a category

$$\mathsf{Coalg}\, H\,.$$

Coalgebras are dual to algebras in the sense that the dual of $\mathsf{Coalg}\, H$ is the category of algebras for the functor $H^{\mathrm{op}} : \mathcal{A}^{\mathrm{op}} \longrightarrow \mathcal{A}^{\mathrm{op}}$:

$$(\mathsf{Coalg}\, H)^{\mathrm{op}} = \mathsf{Alg}\, H^{\mathrm{op}}\,.$$

(ii) Dually to the situation in algebra, in coalgebra the concept of a *final* (or, *terminal*) coalgebra plays a major rôle. An H-coalgebra (T, t) is final if for any H-coalgebra (A, a) there exists a unique coalgebra homomorphism

$$[\![a]\!] : A \longrightarrow T\,.$$

This notation is to indicate that a final coalgebra is used to provide semantics of computations in dynamical systems. In fact, we shall see in the Examples 2.29 how $[\![a]\!]$ assigns to an element of A thought of as a state of a system its semantics as an element of the terminal coalgebra. This element can often be thought of as the *behaviour* of the state, i. e., the information which can be observed of the state using (repeatedly) all the observations provided by the coalgebra structure a.

Notice that a final H-coalgebra (T, t) is a terminal object of $\mathsf{Coalg}\, H$, and as such it is uniquely determined up to isomorphism.

Dualizing Lambek's Lemma 2.20 we obtain the following

Lemma 2.28. *If (T, t) is a final coalgebra, then t is an isomorphism.*

Consequently, a final H-coalgebra is the greatest fixed point of H, or the greatest solution of the domain equation

$$X \simeq HX\,.$$

Examples 2.29.

(i) Consider the functor given by $HX = X + 1$ on sets. Coalgebras $\alpha : A \longrightarrow A + 1$ are just partial unary algebras and coalgebra homomorphisms are the strict algebra homomorphisms $h : (A, \alpha) \longrightarrow (B, \beta)$, i. e.,

$$\beta \text{ is defined on } h(a) \text{ if and only if } \alpha \text{ is defined on } a.$$

Of course, we can also view (A, α) as a deterministic automaton (with unary input) with halting states, i. e., states which do not have a successor state. Usually, one writes

$$a \longrightarrow a' \qquad \text{and} \qquad a{\downarrow}$$

if a can make a state transition to a', or if a is a halting state, respectively. The coalgebra homomorphisms are functions $h : A \longrightarrow B$ preserving halting states and transitions:

(a) $a{\downarrow}$ implies $h(a){\downarrow}$, and

(b) $a \longrightarrow a'$ implies $h(a) \longrightarrow h(a')$

These functions are called simulations in automata theory. Notice that due to the determinism preservation of halting states and transitions implies their reflection by h.

A final H-coalgebra is carried by

$$\overline{\mathbb{N}} = \mathbb{N} + \{\infty\}$$

with the structure map $\mathsf{pred} : \overline{\mathbb{N}} \longrightarrow \overline{\mathbb{N}} + 1$

$$\mathsf{pred}(n) = \begin{cases} n-1 & \text{if } n \in \mathbb{N} \setminus \{0\} \\ * & \text{if } n = 0 \\ \infty & \text{if } n = \infty \end{cases}$$

where $*$ denotes the unique element of 1. For a coalgebra (A, α) the uniquely determined map $[\![\alpha]\!] : A \longrightarrow \overline{\mathbb{N}}$ assigns to every state a of A its behaviour. More precisely, $[\![\alpha]\!]$ assigns $*$ to any halting state, and to any other state a the number of steps the automaton A can make from a before a halting state is reached—this number may be infinity.

The universal property can also be understood in terms of a definition and a proof principle called corecursion and coinduction, respectively. We do not give any details of this here. The interested reader should consult the aforementioned introductory texts [JR, R_2, Gu, A_2].

(ii) Streams. Coalgebras for the functor given by $HX = S \times X$ for a set S are deterministic automata with output in S. Indeed, a coalgebra structure $\alpha = \langle o, s\rangle : A \longrightarrow S \times A$ gives for each state $a \in A$ a pair consisting of the output of state a and its successor state. Coalgebra homomorphisms are, again, simulations, i. e., maps preserving (and therefore reflecting) transitions and output. In fact, h is a coalgebra homomorphism from $(A, \langle o, s\rangle)$ to $(A', \langle o', s'\rangle)$ if and only if $o'(h(a)) = o(a)$ and $s'(h(a)) = h(s(a))$. A final coalgebra for H is carried by the set of streams (i. e., infinite sequences) S^ω with the head and tail function as structure

$$\langle \mathsf{hd}, \mathsf{tl}\rangle : S^\omega \longrightarrow S \times S^\omega .$$

For a coalgebra (A, α) the mapping $[\![\alpha]\!] : A \longrightarrow S^\omega$ assigns to a state of A the stream of outputs produced by the computation that starts at this state.

(iii) Deterministic Automata. The application of the theory of coalgebras to automata theory has been studied by Jan Rutten [R_1]. Here we will just indicate how automata can be understood as coalgebras. Deterministic automata with an input alphabet Σ are usually presented by a set of states Q, a next state function $\delta : Q \times \Sigma \longrightarrow Q$ and a subset $F \subseteq Q$ of final states, equivalently, a function $f : Q \longrightarrow 2$ where $2 = \{0, 1\}$ with $f(q) = 1$ if and only if $q \in F$. We disregard initial states since they are inessential when automata are treated as coalgebras. Currying δ, we get a function $\overline{\delta} : Q \longrightarrow Q^\Sigma$ defined by $\overline{\delta}(q) = \delta(q, -) : \Sigma \longrightarrow Q$. Now an automaton is presented by one function

$$\alpha = \langle \delta, f\rangle : Q \longrightarrow Q^\Sigma \times 2 ,$$

thus, it is a coalgebra for the functor given by $HX = X^\Sigma \times 2$. The coalgebra homomorphisms are the functional bisimulations: h is a coalgebra homomorphism from $(Q, \langle \delta, f\rangle)$ to $(Q', \langle \delta', f'\rangle)$ if

(a) q is final in Q if and only if $h(q)$ is final in Q', i. e., $f'(h(q)) = f(q)$, and

(b) h preserves (and therefore reflects) state transitions, i. e.,

$$h(\delta(q, s)) = \delta'(h(q), s)$$

for all $q \in Q$ and $s \in \Sigma$.

A final coalgebra is carried by the set

$$T = \mathcal{P}(\Sigma^*)$$

of all formal languages over Σ with the following structure $\tau : \langle \delta, f\rangle : T \longrightarrow T^\Sigma \times 2$:

$$\begin{aligned} f(L) &= 1 \qquad \text{if the empty word lies in } L \\ \delta(L) &= (s \longmapsto L_s = \{\, w \in \Sigma^* \mid sw \in L \,\}) \end{aligned}$$

Now for an automaton considered as a coalgebra (A, α) the map $[\![\alpha]\!] : A \longrightarrow T$ assigns to each state a the language the automaton accepts when we choose a as the initial state of the automaton. This is the reason why we said that initial states are inessential here. The map $[\![\alpha]\!]$ summarizes the semantics of the automaton regardless of what state is the initial one.

(iv) Finitely branching nondeterministic automata (with unary input) are simply coalgebras for the finite power set functor $\mathcal{P}_{\text{fin}}$. Equivalently, these are finitely branching directed graphs. However, it may be misleading to think about $\mathcal{P}_{\text{fin}}$-coalgebras that way because coalgebra homomorphisms are stronger than the usual graph homomorphisms. In fact, h is a coalgebra homomorphism from a graph G to a graph H considered as $\mathcal{P}_{\text{fin}}$-coalgebras if and only if h is a graph homomorphism that reflects edges: if $h(a) \longrightarrow b$ is an edge in H then there exists an edge $a \longrightarrow a'$ in G with $h(a') = b$. Switching back to the system theoretic view of $\mathcal{P}_{\text{fin}}$-coalgebras the homomorphisms are precisely the functional bisimulations, i. e., the maps that preserve and reflect transitions.

A terminal $\mathcal{P}_{\text{fin}}$-coalgebra is the set of all (finite and infinite) strongly extensional finitely branching trees where the coalgebra structure assigns to a tree the finite set of its maximum proper subtrees, see [W$_2$].

(v) Labelled transition systems (lts) are an important tool in the semantics of process algebra. An lts is a triple $(L, \longrightarrow, \mathsf{Act})$ where Act is a set of (atomic) actions, L is a set of states and $\longrightarrow$ is a family of relations $\xrightarrow{a}$ on L indexed by elements a of Act. As usual, one writes

$$s \xrightarrow{a} s'$$

if s and s' are related by $\xrightarrow{a}$, and we think of this as a state transition from s to s' "consuming" a. Of course, every transition system can be summarized by a map

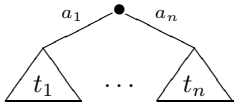
$$\alpha : L \longrightarrow \mathcal{P}(\mathsf{Act} \times L)\,,$$

where $(a, s') \in \alpha(s)$ if and only if $s \xrightarrow{a} s'$, i. e., lts are precisely the coalgebras for the endofunctor $\mathcal{P}(\mathsf{Act} \times -)$ of Set. Coalgebra homomorphism are functional (strong) bisimulations, i. e., those maps that preserve and reflect transitions: a map $h : L \longrightarrow L'$ between state sets of two lts is a coalgebra homomorphism if and only if

(a) $s \xrightarrow{a} t$ implies $h(s) \xrightarrow{a} h(t)$, and

(b) if $h(s) \xrightarrow{a} t'$, then there exists a state t in L with $s \xrightarrow{a} t$ and $h(t) = t'$.

It is clear that there can be no final coalgebra as the functor $\mathcal{P}(\mathsf{Act} \times -)$ does not have fixed points. If however, we were to put a bound on the branching breadth of lts this situation can be remedied. For example, finitely branching transition systems are coalgebras for $\mathcal{P}_{\text{fin}}(\mathsf{Act} \times -)$, and this functor has a final coalgebra in Set. It is carried by the set T of all finitely branching strongly extensional trees with edges labelled in Act. The coalgebra structure $T \longrightarrow \mathcal{P}_{\text{fin}}(\mathsf{Act} \times T)$ assigns to any tree



the finite set of pairs (a_i, t_i), $i = 1, \ldots, n$. The map $[\![\alpha]\!] : L \longrightarrow T$ assigns to every state its behaviour in the following way: first take the tree unfolding t of the state according to the transitions of L. Since this may not be strongly extensional one identifies bisimilar children of any node in t so that a strongly extensional tree is obtained. Notice that here the notion of bisimulation takes into account the labels of edges. We leave the technical details to the reader.

(vi) Infinite Σ-trees. Let Σ be a signature of operation symbols with prescribed arity and form the polynomial functor $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$. The H_Σ-coalgebras are automata with inputs from the set of natural numbers and with output from Σ, where for each state the maximum possible input number is the arity of the output of that state. More precisely, if $\alpha : A \longrightarrow H_\Sigma A$ is a coalgebra we can think of A as the set of states and α assigns to every state a its reaction $(\sigma, a_1, \ldots, a_n)$ where $\sigma \in \Sigma_n$, for some natural number n is the output of a and $a_1, \ldots, a_n$ are the successor states corresponding to the transitions that will occur if at state a we input a number $k = 1, \ldots, n$ into the automaton. The coalgebra homomorphisms are the usual automata simulations, and we leave it to the reader to work this out. Recall that the initial Σ-algebra is formed by all finite Σ-trees, see Example 2.17(iv). If we take instead all (finite and infinite) Σ-trees we get the final coalgebra T_Σ whose structure map is the inverse of tree-tupling which is sometimes called "parsing". It assigns to a Σ-tree t whose root is labelled by an n-ary operation symbol σ the $(n+1)$-tuple $(\sigma, t_1, \ldots, t_n)$ where t_i, $i = 1, \ldots n$, are the maximum proper subtrees of t.

(vii) Final coalgebras for finitary set functors. Suppose that H is a finitary endofunctor of Set described as a quotient $\varepsilon : H_\Sigma \longrightarrow H$ as in Example 2.17(vii).

In joint work with Jiří Adámek [AM$_1$] we have proved that the final H-coalgebra T is given by the quotient $T_\Sigma/\!\sim^*$ where $\sim^*$ is the following congruence: for every Σ-tree t denote by $\partial_n t$ the finite tree obtained by cutting t at level n and labelling all leaves at that level by some symbol $\perp$ not from Σ. Then we have $s \sim^* t$ for two Σ-trees s and t if and only if for all $n < \omega$, $\partial_n s$ can be obtained from $\partial_n t$ by finitely many applications of the basic equations describing ε, more shortly, $s \sim^* t$ if and only if $\partial_n s \sim \partial_n t$ for all natural numbers n, where $\sim$ is the congruence of Example 2.17(vii). For example, for the endofunctor given by $HX = \{\, \{\, x, y, \,\} \mid x, y \in X \,\} + 1$ of Example 2.17(vii) the terminal coalgebra is the coalgebra of all unordered binary trees.

Now we will list some results from the general theory of coalgebras that we will later frequently use. Dually to Theorem 2.19, we have

Theorem 2.30. *The forgetful functor $U : \mathsf{Coalg}\, H \longrightarrow \mathcal{A}$ creates all colimits and all limits that H preserves.*

Remark 2.31. Of course, dualizing the concept of a free algebra yields the important concept of a cofree coalgebra. While this plays a major rôle in the general theory of coalgebras we will not mention anything about it here since it plays no rôle for what we are about to present.

Just as initial (or, more generally, free) algebras can be constructed inductively as a colimit of a chain, a final H-coalgebra can be constructed dually as a limit of an (op-)chain. Again, let us first consider an endofunctor H of Set. Dual to the initial algebra chain (i. e., the free algebra chain for the case $X = 0$, see 2.22(i)) we define by transfinite induction the following chain indexed by all ordinal numbers:

$$\begin{array}{lrcll}
\text{Initial step:} & T_0 & = & 1, & t_{1,0} \equiv H1 \xrightarrow{\;!\;} 1, \\
\text{Isolated step:} & T_{i+1} & = & HT_i, & t_{i+1,j+1} \equiv HT_i \xrightarrow{Ht_{i,j}} HT_j \\
\text{Limit step:} & T_j & = & \lim\limits_{i<j} T_i & \text{with limit cone } t_{j,i} : T_j \longrightarrow T_i, \quad i < j,
\end{array}$$

the connecting map $t_{j+1,j}$ is uniquely determined by the commutativity of the squares

$$\begin{array}{ccc}
T_{i+1} = HT_i & \xleftarrow{Ht_{j,i}} & HT_j = T_{j+1} \\
{\scriptstyle t_{i+1,i}}\downarrow & & \vdots\,{\scriptstyle t_{j+1,j}} \\
T_i & \xleftarrow[t_{j,i}]{} & T_j
\end{array} \qquad i < j\,.$$

This chain is said to *converge* if $t_{i+1,i}$ is an isomorphism for some ordinal number i.

Theorem 2.32. ([AK$_3$])
Let H be an endofunctor of Set. Then the above chain T_i converges if and only if H has a final coalgebra. Moreover, if i is an ordinal number such that $t_{i+1,i}$ is an isomorphism, then $(T_i, t_{i+1,i}^{-1})$ is a final coalgebra.

This result does not indicate after how many steps a final coalgebra (if it exists) can be obtained. However, dualizing Theorem 2.25 we see that for a λ-continuous functor, i. e., a functor that preserves limits of λ^{op}-chains, a final coalgebra is obtained in λ steps.

Corollary 2.33. *For a λ-continuous functor H, the above final coalgebra chain T_i converges after λ steps and $(T_\lambda, t_{\lambda+1,\lambda}^{-1})$ is a final H-coalgebra.*

Again, this last result holds in every category $\mathcal{A}$ having limits of α^{op}-chains, $\alpha \leq \lambda$, and for every λ-continuous functor on $\mathcal{A}$.

Unfortunately, this result is not as satisfactory as Theorem 2.25 before since many everyday functors are continuous only for large enough λ, for example, the finite power set functor is not ω-continuous but only ω_1-continuous where ω_1 is the first uncountable cardinal. However, for finitary set functors James Worrell [W$_1$, W$_2$] has proved that the final coalgebra construction converges after $\omega \cdot 2$ steps.

Theorem 2.34. *If H is a finitary endofunctor of Set, then the above final coalgebra chain converges after $\omega \cdot 2$ steps, i. e., $(T_{\omega 2}, t_{\omega 2+1,\omega 2}^{-1})$ is a final H-coalgebra.*

In fact, this result was proved for all accessible (equivalently, bounded) set functors. More precisely, if H is a λ-accessible set functor, see Remark 2.13, then the final coalgebra chain converges after $\lambda \cdot 2$ steps.

Example 2.35. To illustrate the result of Theorem 2.34 let us consider again the finite power set functor $\mathcal{P}_{\text{fin}}$ of Set. The sets T_i, $i < \omega$, consist of all finite strongly extensional trees of depth i. One can show that the limit T_ω can be described as the set of all countable unordered trees modulo the following equivalence $\approx$: we have $s \approx t$ if for all natural numbers n the strongly extensional quotients of the trees obtained by cutting s and t, respectively, at level n agree. Then for every $i < \omega$, $T_{\omega+i} \hookrightarrow T_\omega$ is the subset given by equivalence classes of $\approx$ given by all those unordered trees which are finitely branching up to level i, i. e., every node that can be reached from the root by a path with less than i steps has only finitely many children. Thus, in the second limit step we get the intersection of all $T_{\omega+i}$, $i < \omega$, whence $T_{\omega+\omega}$ is the set of all equivalence classes of $\approx$ given by finitely branching trees. One readily proves that the finitely branching strongly extensional trees form a set of representatives of the equivalence classes of $T_{\omega+\omega}$ as desired.

2.4 Monads

In this subsection we recall basic facts about monads, a categorical notion which is central to our work. Formally, monads are endofunctors $M : \mathcal{A} \longrightarrow \mathcal{A}$ with extra structure. One may think of this structure as an abstract way of expressing that MX consists of syntactic objects of some kind, e. g., terms over a signature with variables from a set X. In fact, the axioms of monads capture the essence of substitution of other syntactic objects for variables. Monads are also a tool that allows to capture algebras with operations that satisfy equations, e. g., monoids, groups, etc. The action of the monad on objects can then be thought of as assigning a free algebra to an object of generators. Actually, monads on Set are equivalent to algebraic theories in the sense of Lawvere [La] and Linton [Li$_1$].

Definition 2.36. A *monad* is a triple

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

consisting of a functor $M : \mathcal{A} \longrightarrow \mathcal{A}$ and two natural transformations $\eta : Id \longrightarrow M$, called the *unit*, and $\mu : MM \longrightarrow M$, called the *multiplication*, such that the following diagrams

$$\begin{array}{ccccc} M & \xrightarrow{\eta M} & MM & \xleftarrow{M\eta} & M \\ & \searrow & \downarrow{\scriptstyle \mu} & \swarrow & \\ & & M & & \end{array} \qquad \begin{array}{ccc} MMM & \xrightarrow{\mu M} & MM \\ {\scriptstyle M\mu}\downarrow & & \downarrow{\scriptstyle \mu} \\ MM & \xrightarrow[\mu]{} & M \end{array}$$

commute. The corresponding equations are often called left and right unit laws, and associativity.

A *finitary monad* is a monad $\mathbb{M}$ whose underlying functor is finitary.

Remark 2.37. An equivalent presentation of a monad is as a *Kleisli-triple*

$$(M, \eta, (\widehat{-}))\,,$$

where M is an object assignment of $\mathcal{A}$, η a family of morphisms $\eta_X : X \longrightarrow MX$ indexed by objects of $\mathcal{A}$ and $(\widehat{-})$ is a function assigning to any $f : X \longrightarrow MY$ a morphism $\widehat{f} : MX \longrightarrow MY$ subject to three axioms: for any $f : X \longrightarrow MY$ and $g : Y \longrightarrow MZ$ we have

(i) $\widehat{f} \cdot \eta_X = f$ (extension)

(ii) $\widehat{\eta_X} = id$ (unit)

(iii) $\widehat{\widehat{f} \cdot g} = \widehat{f} \cdot \widehat{g}$ (composition)

It is an easy exercise to check that Kleisli-triples and monads are equivalent presentations, see [Ma], Sec. 3, Ex. 12. In fact, any Kleisli-triple gives a monad: let $\mu_X = \widehat{id_{MX}}$ and check naturality of η and μ and the three monad laws. Conversely, every monad yields a Kleisli-triple: let $\widehat{f} = \mu_Y \cdot Mf$ and check the above three laws.

Examples 2.38.

(i) The identity functor on every category is a monad with $\eta = \mu = id$.

(ii) On Set the list monad $((\_)^*, \eta, \mu)$ is given as follows: to a set X assign the set of lists X^* of elements of X, the unit is given by $\eta_X : x \longmapsto \langle x \rangle$, and multiplication is flattening: to a list $\langle l_1, \ldots, l_n \rangle$ of lists over X, μ_X assigns the list obtained by concatenating the constituent lists l_i, $i = 1, \ldots, n$.

(iii) For every varietor H, the assignment $X \longmapsto FX$ of a free algebra on an object of generators gives a monad (F, η, μ) with the unit given by the universal arrows $\eta_X : X \longrightarrow FX$ and the components of the multiplications obtained as the unique homomorphic extensions $\mu_X : FFX \longrightarrow FX$ of the identity of FX.

(iv) The power set functor $\mathcal{P} : \mathsf{Set} \longrightarrow \mathsf{Set}$ carries a the structure of a monad: for a set X the unit is $\eta_X : x \longrightarrow \{ x \}$ and the multiplication $\mu_X : \mathcal{P}\mathcal{P}X \longrightarrow \mathcal{P}X$ assigns to a subset of $\mathcal{P}X$ the union of its elements.

(v) Let X be a poset considered as a category. Then a monad is precisely the same as a closure operator. In fact, $M : X \longrightarrow X$ is a monotone map with $x \leq Mx$ (due to the unit), $M \cdot M = M$ (due to the multiplication and the unit laws).

(vi) Let Σ be a finitary signature and let E be a finite set of equations. This defines a finitary variety $\mathcal{V}$ of Σ-algebras. The assignment $X \longmapsto MX$ of a free algebra in $\mathcal{V}$ to the set X of generators is a monad. The unit is given by the universal arrows $\eta_X : X \longrightarrow MX$. The multiplication μ_X is, again, given by the unique homomorphic extension of id_{MX}.

Definition 2.39. A monad morphism between monads (M, η, μ) and (M', η', μ') is a natural transformation $m : M \longrightarrow N$ such that the following diagrams

$$\begin{array}{ccc} Id & \xrightarrow{\eta} & M \\ & \searrow^{\eta'} & \downarrow m \\ & & M' \end{array} \qquad \begin{array}{ccc} MM & \xrightarrow{m*m} & M'M' \\ {\scriptstyle\mu}\downarrow & & \downarrow{\scriptstyle\mu'} \\ M & \xrightarrow[m]{} & M' \end{array}$$

commute. Notice that we denote here by $*$ the parallel composition of natural transformations, i. e., $m * m = Mm \cdot mM = mM' \cdot M'm$.

Notation 2.40. Just as (finitary) endofunctors on a category $\mathcal{A}$ form the categories $\mathsf{Fin}[\mathcal{A}, \mathcal{A}]$ and $[\mathcal{A}, \mathcal{A}]$, respectively, (finitary) monads and monad morphisms form categories

$$\mathsf{M}(\mathcal{A}) \qquad \text{and} \qquad \mathsf{FM}(\mathcal{A})\,,$$

respectively.

Remark 2.41. There is, of course, a canonical notion of morphisms of Kleisli-triples. A morphism between Kleisli-triples from $(M, \eta, (\widehat{-}))$ to $(M', \eta', (\overline{-}))$ is a family of morphisms $m_X : MX \longrightarrow M'X$ indexed by objects such that for all objects X and Y and morphisms $f : X \longrightarrow MY$ the diagrams

$$\begin{array}{ccc} Id & \xrightarrow{\eta} & M \\ & \searrow^{\eta'} & \downarrow m \\ & & M' \end{array} \qquad \begin{array}{ccc} MX & \xrightarrow{m_X} & M'X \\ {\scriptstyle\widehat{f}}\downarrow & & \downarrow{\scriptstyle\overline{m_Y \cdot f}} \\ MY & \xrightarrow[m_Y]{} & M'Y \end{array}$$

commute. It is easy to prove that the category of Kleisli-triples on $\mathcal{A}$ is isomorphic to $\mathsf{M}(\mathcal{A})$.

2.4.1 Free Monads and Second-Order Substitution

We have seen in Example 2.38(iii) that for a varietor H the assignment of a free algebra to an object gives a monad (F, η, μ). Michael Barr [B_1] proved that this is actually a *free monad* on H with the universal arrow given by the natural transformation

$$\kappa \equiv H \xrightarrow{H\eta} HF \xrightarrow{\varphi} F \tag{2.19}$$

where φ expresses the structures of the free algebras. Conversely, a functor which has a free monad is a varietor whenever the base category is "good", see [Ma, B_2] and the following theorem.

Theorem 2.42.

(i) Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be a varietor. The free algebra monad (F, η, μ) is a free monad on H, i. e., for each monad (F', η', μ') and for each natural transformation $\lambda : H \longrightarrow F'$ there exists a unique monad morphism $\overline{\lambda} : F \longrightarrow F'$ such that $\overline{\lambda} \cdot \kappa = \lambda$.

(ii) Conversely, let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be a functor on a complete and well-powered category $\mathcal{A}$. If there exists a free monad on H, then H is a varietor, and for every object X, FX carries a free H-algebra on X.

Remark 2.43. The first part of this result means that the forgetful functor $U : \mathsf{M}(\mathcal{A}) \longrightarrow [\mathcal{A}, \mathcal{A}]$ has a universal arrow at every varietor. However, U may fail to have a left-adjoint. If we restrict the codomain to finitary functors, then we get an adjoint situation

$$\mathsf{FM}(\mathcal{A}) \underset{\longrightarrow}{\overset{\longleftarrow}{\bot}} \mathsf{Fin}[\mathcal{A}, \mathcal{A}]$$

since for each finitary functor H a free monad on H is finitary.

Example 2.44. (Second-order substitution of finite Σ-trees)
As polynomial functors H_Σ are varietors, the assignment of a free Σ-algebra $F_\Sigma X$ (of all finite Σ-trees over X) to a set X yields a monad F_Σ. By Theorem 2.42, F_Σ is a free monad on H_Σ.

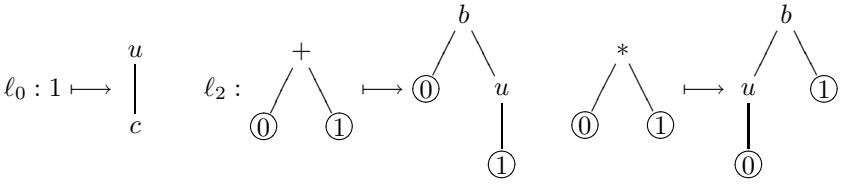
It is well-known that the freeness of the algebras $F_\Sigma X$ boils down to substitution of Σ-trees for variables. It seems to be less well-known that the universal property of the monad F_Σ specializes to *second-order substitution*, i. e., substitution of trees for operation symbols, see [C]. Before we make this precise we need to recall an important property of signatures and polynomial functors.

Let Σ be a signature, i. e., $\Sigma : \mathbb{N} \longrightarrow \mathsf{Set}$ is a functor where $\mathbb{N}$ is regarded as a discrete category. Let $J : \mathbb{N} \longrightarrow \mathsf{Set}$ be the functor which maps a natural number n to the set $\{\, 0, \ldots, n-1 \,\}$. Recall that the functor $(\_) \cdot J : [\mathsf{Set}, \mathsf{Set}] \longrightarrow [\mathbb{N}, \mathsf{Set}]$ of restriction to $\mathbb{N}$ has a left-adjoint $\mathrm{Lan}_J(\_)$, i. e. the functor assigning to a signature its left Kan extension along J. Since $\mathbb{N}$ is a discrete category, the usual coend formula for computing left Kan extensions, see e. g. [ML], Theorem X.4.1, specializes to the coproduct in (2.16) in Example 2.17(v). That is, $\mathrm{Lan}_J(\Sigma)$ is the polynomial functor H_Σ. By virtue of the adjunction $\mathrm{Lan}_J(\_) \dashv (\_) \cdot J$ there is for every signature Σ and every endofunctor G of Set a bijection between natural transformations $\Sigma \longrightarrow G \cdot J$ and natural transformation $H_\Sigma \longrightarrow G$, and this bijection is natural in Σ and G.

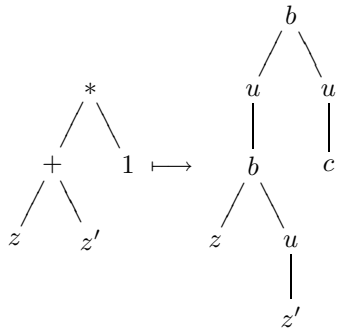
Now let Γ be another signature. Consider each $\sigma \in \Sigma_n$ as a flat tree in n distinct variables. A second-order substitution gives an "implementation" to each such σ as a finite Γ-tree in the same n variables. We model this by a natural transformation $\ell : \Sigma \longrightarrow F_\Gamma \cdot J$, i. e., a family or maps $\ell_n : \Sigma_n \longrightarrow F_\Gamma\{\, 0, \ldots n-1 \,\}$, $n \in \mathbb{N}$. As we have seen above, this gives rise to a natural transformation $\lambda : H_\Sigma \longrightarrow F_\Gamma$. Thus, from Theorem 2.42 we get a monad morphism $\overline{\lambda} : F_\Sigma \longrightarrow F_\Gamma$. For every set X of variables its action is that of second-order substitution, i. e., $\overline{\lambda}_X$ replaces every Σ-symbol in a tree t from $F_\Sigma X$ by its implementation according to ℓ. More precisely, let $t = \sigma(t_1, \ldots, t_n)$ with $\sigma \in \Sigma_n$ and let $t'(x_1, \ldots x_n) \in F_\Gamma X$ be the implementation of σ, i. e., $\ell_n(\sigma) = t'(0, \ldots, n-1)$. Then we have

$$\overline{\lambda}_X(t) = t'(\overline{\lambda}_X(t_1), \ldots, \overline{\lambda}_X(t_n)).$$

For example, suppose that Σ consists of two binary symbols $+$ and $*$ and a constant 1, and Γ consists of a binary symbol b, a unary one u and a constant c. Furthermore, let λ be given by $\ell : \Sigma \longrightarrow F_\Gamma \cdot J$ as follows:



and else ℓ_n is the unique map from the empty set. For the set $Z = \{\, z, z' \,\}$, the second-order substitution morphism $\overline{\lambda}_Z$ acts for example as follows:



2.4.2 Algebras for a Monad

It is well-known that for any adjoint situation

$$\mathcal{B} \underset{U}{\overset{F}{\underset{\longrightarrow}{\overset{\longleftarrow}{\bot}}}} \mathcal{A}\,, \tag{2.20}$$

with unit $\eta : Id \longrightarrow UF$ and counit $\varepsilon : FU \longrightarrow Id$ we obtain a monad

$$(UF, \eta, U\varepsilon F)$$

on $\mathcal{A}$. Conversely, given a monad $\mathbb{M} = (M, \eta, \mu)$ on $\mathcal{A}$ there are two ways to get an adjoint situation which induces the monad $\mathbb{M}$. The first one is via the *Kleisli category* $\mathcal{A}_{\mathbb{M}}$. Objects of $\mathcal{A}_{\mathbb{M}}$ are those of $\mathcal{A}$ and for morphisms we have $\mathcal{A}_{\mathbb{M}}(X, Y) = \mathcal{A}(X, MY)$ with identities $\eta_X : X \longrightarrow MX$ and composition defined as follows: for $f : X \multimap Y$ and $g : Y \multimap Z$ in $\mathcal{A}_{\mathbb{M}}$, i. e., we have $f : X \longrightarrow MY$ and $g : Y \longrightarrow MZ$ in $\mathcal{A}$, we let

$$g \circ f \equiv X \xrightarrow{f} MY \xrightarrow{Mg} MMZ \xrightarrow{\mu_Z} MZ\,.$$

There is a functor $U_{\mathbb{M}} : \mathcal{A}_{\mathbb{M}} \longrightarrow \mathcal{A}$ given by

$$U_{\mathbb{M}}(f : X \multimap Y) = MX \xrightarrow{Mf} MMY \xrightarrow{\mu_Y} MY$$

with a left adjoint $F_{\mathbb{M}} : \mathcal{A} \longrightarrow \mathcal{A}_{\mathbb{M}}$ which is identity on objects and is defined on morphisms by

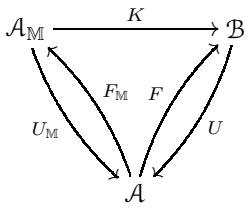
$$F_{\mathbb{M}}(f : X \longrightarrow Y) = X \xrightarrow{f} Y \xrightarrow{\eta_Y} MY\,.$$

It is an easy task to check that there is indeed an adjoint situation

$$\mathcal{A}_{\mathbb{M}} \underset{U_{\mathbb{M}}}{\overset{F_{\mathbb{M}}}{\underset{\longrightarrow}{\overset{\longleftarrow}{\bot}}}} \mathcal{A} \tag{2.21}$$

and that the monad induced by this adjunction is $\mathbb{M}$.

Moreover, the adjunction (2.21) is the smallest one giving $\mathbb{M}$ in the following sense: for any adjunction (2.20) defining $\mathbb{M}$ there exists a unique functor $K : \mathcal{A}_{\mathbb{M}} \longrightarrow \mathcal{B}$ with $U \cdot K = U_{\mathbb{M}}$ and $F = K \cdot F_{\mathbb{M}}$:



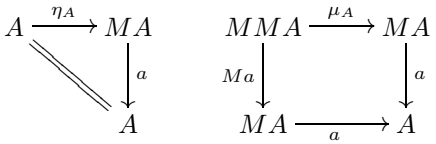
In fact, K is defined by

$$KX = FX \qquad \text{and} \qquad K(f : X \multimap Y) = \widehat{f} : FX \longrightarrow FY\,,$$

where $\widehat{f}$ is the uniquely determined morphism with $U\widehat{f} \cdot \eta_X = f : X \longrightarrow UFY$.

The second way to obtain an adjoint situation describing a given monad is via its Eilenberg-Moore category.

Definition 2.45. Let $\mathbb{M} = (M, \eta, \mu)$ be a monad on $\mathcal{A}$. An *Eilenberg-Moore algebra* for $\mathbb{M}$ is a pair (A, a) consisting of a carrier A and a structure $a : MA \longrightarrow A$ subject to the axioms



which are often called *unit law* and *associativity*, respectively.

A homomorphism of Eilenberg-Moore algebras is a homomorphism of the underlying algebras for the functor M.

Remark 2.46. Eilenberg-Moore algebras and their homomorphisms form a category $\mathcal{A}^{\mathbb{M}}$ and we have an obvious forgetful functor

$$U^{\mathbb{M}} : \mathcal{A}^{\mathbb{M}} \longrightarrow \mathcal{A}\,,$$

which always has a left-adjoint $F^{\mathbb{M}} : A \longrightarrow \mathcal{A}^{\mathbb{M}}$ defined by

$$FX = (MX, \mu_X) \qquad \text{and} \qquad F(f : X \longrightarrow Y) = Mf\,.$$

Again, the adjunction

$$\mathcal{A}^{\mathbb{M}} \underset{U^{\mathbb{M}}}{\overset{F^{\mathbb{M}}}{\underset{\longrightarrow}{\overset{\longleftarrow}{\bot}}}} \mathcal{A} \tag{2.22}$$

induces the monad $\mathbb{M}$.

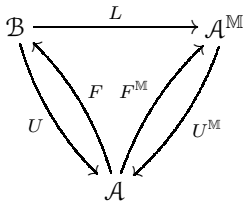
The categories $\mathcal{A}^{\mathbb{M}}$ of algebras for a monad have "algebraic flavour" as witnessed for example by the following result.

Theorem 2.47. *The forgetful functor $U^{\mathbb{M}}$ creates all limits and all colimits that are preserved by M.*

In particular, if $\mathcal{A}$ is an lfp category and $\mathbb{M}$ is finitary then $U^{\mathbb{M}}$ creates filtered colimits. Moreover, we have the following result, see [AR], 2.78.

Theorem 2.48. *Let $\mathbb{M}$ be a finitary monad on an lfp category. Then $\mathcal{A}^{\mathbb{M}}$ is lfp.*

Now suppose we have an adjoint situation (2.20) with its induced monad $\mathbb{M}$. Then the adjunction (2.22) is the largest one inducing $\mathbb{M}$ in the following sense: for any adjunction (2.20) inducing $\mathbb{M}$ there exists a unique functor $L : \mathcal{B} \longrightarrow \mathcal{A}^{\mathbb{M}}$ satisfying $U^{\mathbb{M}} \cdot L = U$ and $F^{\mathbb{M}} = L \cdot F$:



In fact, L is defined by

$$LB = (UB, U\varepsilon_B : MUB = UFUB \longrightarrow UB) \qquad \text{and} \qquad Lf = Uf\,.$$

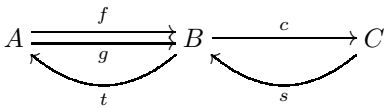
One may think of L as measuring the degree of "algebraicness" of $\mathcal{B}$ over $\mathcal{A}$. In fact, if L is an isomorphism one can regard the category $\mathcal{B}$ as a category of algebras.

Definition 2.49. A functor $U : \mathcal{B} \longrightarrow \mathcal{A}$ is called *monadic* provided that

(i) U has a left adjoint $F : \mathcal{A} \longrightarrow \mathcal{B}$

(ii) the comparison functor $L : \mathcal{B} \longrightarrow \mathcal{A}^{\mathbb{M}}$, where $\mathbb{M}$ is the monad induced by the adjunction $F \dashv U$, is an isomorphism of categories.

Monadic functors $U : \mathcal{B} \longrightarrow \mathcal{A}$ can be characterized with respect to the creation of certain coequalizers by U. Recall that a *split coequalizer* of a parallel pair $f, g : A \longrightarrow B$ is a morphism $c : B \longrightarrow C$ so that we have two more morphisms s and t as in the diagram



so that $c \cdot f = c \cdot g$, $c \cdot s = id$, $s \cdot c = f \cdot t$ and $g \cdot t = id$. In fact, it is easy to check that c is then a coequalizer of f and g. Recall further, that a colimit is called *absolute* provided that it is preserved by every functor. The proof of the following Theorem can be found in [ML].

Theorem 2.50. (Beck's Theorem)
Let $F \dashv U : \mathcal{B} \longrightarrow \mathcal{A}$ be an adjunction with an induced monad $\mathbb{M}$. Then the following are equivalent:

(i) U is monadic

(ii) U creates coequalizers which are U-absolute, i. e., coequalizers of those parallel pairs f, g in $\mathcal{B}$ for which Uf, Ug has an absolute coequalizer in $\mathcal{A}$.

(iii) U creates coequalizers of U-split pairs, i. e., pairs of parallel morphisms f, g in $\mathcal{B}$ for which the pair Uf, Ug has a split coequalizer in $\mathcal{A}$

We conclude this subsection with a number of examples.

Examples 2.51.

(i) Recall that a monoid is an algebra with a binary associative operation and a unit of this operation. The category of monoids and their homomorphisms is monadic over Set (i. e., the corresponding forgetful functor is monadic). In fact, this category is isomorphic to $\mathsf{Set}^{\mathbb{M}}$ where $\mathbb{M} = ((\_)^*, \eta, \mu)$ is the list monad of Example 2.38(ii). For every set X, the set X^* together with the operation of concatenation and the empty list is a free monoid on X.

(ii) More generally, every finitary variety is monadic over Set. In fact, recall that a finitary variety $\mathcal{V}$ is a category of algebras for a signature of operations and finitely many equations between these operations. It is not difficult to show that $\mathcal{V}$ is the category of Eilenberg-Moore algebras for the monad given by assigning to each set X (of generators) a free algebra in $\mathcal{V}$ on X.

(iii) For every varietor $H : \mathcal{A} \longrightarrow \mathcal{A}$, the category $\mathsf{Alg}\, H$ is monadic over $\mathcal{A}$. In fact, $\mathsf{Alg}\, H$ is isomorphic to the category $\mathcal{A}^{\mathbb{F}}$ of Eilenberg-Moore algebras for the free monad $\mathbb{F} = (F, \eta, \mu)$ on H. While this is trivial to prove with the help of Theorem 2.50, a direct proof is quite instructive: the isomorphism $L : \mathsf{Alg}\, H \longrightarrow \mathcal{A}^{\mathbb{F}}$ assigns to each algebra (A, a) the algebra $(A, \widehat{a})$ where $\widehat{a} : FA \longrightarrow A$ is the unique homomorphic extension of $id : A \longrightarrow A$. The inverse L^{-1} takes an algebra (A, α) to

$$HA \xrightarrow{\kappa_A} FA \xrightarrow{\alpha} A\,,$$

where $\kappa : H \longrightarrow F$ is the canonical transformation of (2.19). We leave the details to the reader. Notice that the Eilenberg-Moore algebra structure $\widehat{a} : FA \longrightarrow A$ induced by an algebra (A, a) is a generalized version of the computation of finite trees in a Σ-algebra A, see Example 2.23(i).

(iv) The algebras for the power set monad $\mathcal{P}$ on Set are the complete lattices; more precisely, partially ordered sets with all joins, together with the continuous maps, i. e., function preserving all joins.

(v) Finitary monads are algebras. In fact, suppose that $\mathcal{A}$ is an lfp category. Then the forgetful functor

$$U : \mathsf{FM}(\mathcal{A}) \longrightarrow \mathsf{Fin}[\mathcal{A}, \mathcal{A}]$$

is monadic with its left adjoint assigning to any finitary endofunctor H of $\mathcal{A}$ a free monad on H.

3 Iterative and Completely Iterative Algebras

Is this a dagger which I see before me?
William Shakespeare, *Macbeth*, Act 2, Scence 1.

It was the idea of Calvin Elgot [E] to study algebraically the semantics of recursive computations without using the common approach which is based on working with ordered or metrized structures. He introduced iterative theories, and he and his coauthors proved that the free iterative theory over a signature Σ exists [BE] and that it is the theory formed by rational trees, see [EBT]. Recall that a Σ-tree is called *rational* if it has up to isomorphism finitely many subtrees only, see [Gi_2].

Later it has been realized that a much easier approach to iterative theories is via *iterative algebras*. For a signature Σ they were studied by Evelyn Nelson [N]. Her work showed that the free iterative algebras are the algebras of rational trees over a set X, and that this yields a short proof of Elgot's free iterative theories. The first attempts to capture rational trees categorically were [GLM_1] and [AMV_1]. In the categorical approach one replaces signatures by a finitary endofunctor of Set (or more general base categories). In [AMV_1] we gave a categorical construction of the rational monad generated by such an endofunctor H, and we proved that it is characterized as the free iterative monad on H. The authors of [GLM_1] have generalized our construction but did not prove the universal property of the rational monad in their setting. Both papers do not work with iterative algebras. Moreover, our paper [AMV_1] is rather long and technical, and besides we had to put several side conditions on the category and endofunctor we worked with. In Subsection 3.2 below we will introduce and study iterative algebras for every finitary functor H of an lfp category $\mathcal{A}$. We will show that free iterative algebras exist and we give a coalgebraic construction of them. In this way we get a generalization of Nelson's results and later in Section 6 we show that free iterative algebras yield free iterative monads. Thus we obtain a new proof of Elgot's classical result in the special case of a polynomial functor H_Σ of Set. At the same time we have generalized the result and, moreover, our proof is conceptually much simpler then any of the previous ones.

But before that we introduce and study completely iterative algebras in Subsection 3.1, which allow to give an easy approach to completely iterative theories of Elgot et al. In [EBT] the authors proved that the theory of all Σ-trees forms a free completely iterative theory on a signature Σ. We will generalize this classical result in our work. It turns out that complete iterativity of infinite trees is due to the fact that they form a final coalgebra. The result that final coalgebras form a monad that is completely iterative was discovered independently and almost at the same time in work by Lawrence Moss [Mo_1] and by Peter Aczel, Jiří Adámek, Jiří Velebil [AAV]. That final coalgebras form a monad was also proved for (generalized) polynomial endofunctors of an lfp category by Neil Ghani, Christoph Lüth, Federico De Marchi and John Power [GLMP_1, GLMP_2], see also [DM]. In our paper [AAMV] we proved that final coalgebras yield a monad in general, and that this monad is characterized as the free completely iterative monad. More precisely, one works with a category $\mathcal{A}$ with binary coproducts and a functor H having "enough final coalgebras", i. e., for every object Y of $\mathcal{A}$ there exists a final coalgebra TY of $H(\_) + Y$. Then T forms a free completely iterative monad on H. In [M_1] we have added completely iterative algebras to the picture. We proved that for an object mapping T of $\mathcal{A}$ the following three statements are equivalent:

(a) for every object Y, TY is a final coalgebra for $H(\_) + Y$,

(b) for every object Y, TY is a free completely iterative H-algebra on Y, and

(c) T is a free completely iterative monad on H.

We shall establish the equivalence of (a) and (b) in Subsection 3.1 below. This is the first part of our paper [M_1]. Its second part deals with completely iterative monads, which will be studied in Section 6 where we add (c) to the above list of equivalent statements.

3.1 Completely Iterative Algebras

Before we introduce completely iterative algebras in general we would like to illustrate the leading example of completely iterative algebras for a given signature Σ, i. e., for a polynomial functor H_Σ of Set, a bit more.

A Σ-algebra A is called *completely iterative*, if every system

$$x_i \approx t_i, \quad i \in I, \tag{3.23}$$

where I is some (possibly infinite) set, $X = \{ x_i \mid i \in I \}$ is a set of variables and the t_i are terms over $X + A$, none of which is just a single variable, has a unique solution in A. By a *solution* we mean a set

$\{ {x_i}^\dagger \mid i \in I \}$ of elements of A such that the above formal equations (3.23) become actual identities in A when the variables are substituted by the solutions and the terms t_i are interpreted in A, i. e.,

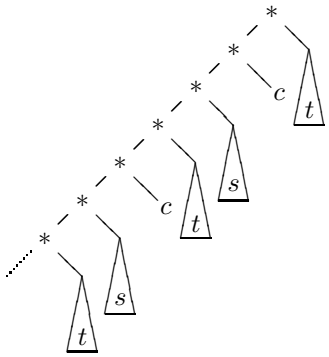
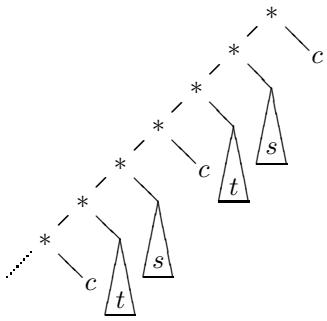
$$ {x_i}^\dagger \equiv \alpha \left(t_i[x_j := {x_j}^\dagger]_{j \in J} \right), \quad i \in I \,, $$

where $\alpha : F_\Sigma A \longrightarrow A$ is the canonical computation of finite Σ-trees in A, see Example 2.23(i).

For example, suppose we have a signature Σ. The algebra $A = T_\Sigma$ of all finite and infinite Σ-trees is completely iterative. For example, let Σ consist of a binary operation symbol $*$ and a constant symbol c. Then the following system

$$ x_1 \approx x_2 * t \qquad x_2 \approx (x_1 * s) * c \tag{3.24} $$

where s and t are some trees in T_Σ has the following solution

$x_1{}^\dagger =$  and $x_2{}^\dagger =$  (3.25)

Observe that it is sufficient to allow for the right-hand sides in (3.24) only so-called *flat terms*, i. e., terms t that are either

$$ t = \sigma(x_1, \ldots, x_n), \qquad \sigma \in \Sigma_n, \quad x_1, \ldots, x_n \in X, $$

or

$$ t \in A. $$

In fact, for every system (3.23) one can give a system with only flat terms as right-hand sides, which has the same solution. This is done by introducing (possibly infinitely many) new variables. For example for the system (3.24) we get the following flat one:

$$ \begin{array}{lcllcl} x_1 & \approx & x_2 * z_1 & z_2 & \approx & x_1 * z_4 \\ x_2 & \approx & z_2 * z_3 & z_3 & \approx & c \\ z_1 & \approx & t & z_4 & \approx & s \end{array} \tag{3.26} $$

Obviously, the solutions ${x_1}^\dagger$ and ${x_2}^\dagger$ are the same trees as before.

Clearly, one can write every system with flat right-hand sides as a single map

$$ e : X \longrightarrow H_\Sigma X + A \,, $$

and a solution is a map $e^\dagger : X \longrightarrow A$ such that the following square

$$ \begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ H_\Sigma X + A & \xrightarrow[H_\Sigma e^\dagger + A]{} & H_\Sigma A + A \end{array} $$

where a denotes the algebra structure of A, commutes. The following are the minimal requirements to formulate these notions.

Assumption 3.1. For the rest of this text we assume that $\mathcal{A}$ is a category with binary coproducts and that $H : \mathcal{A} \longrightarrow \mathcal{A}$ is an endofunctor of $\mathcal{A}$. We shall always denote injections of a coproduct $A + B$ by $\mathsf{inl} : A \longrightarrow A + B$ and $\mathsf{inr} : B \longrightarrow A + B$, and we denote by $\mathsf{can} : HA + HB \longrightarrow H(A + B)$ the canonical morphism $[H\mathsf{inl}, H\mathsf{inr}]$.

Definition 3.2. A morphism $e : X \longrightarrow HX + A$ of $\mathcal{A}$ is called a *flat equation morphism* in (the object of parameters) A. Suppose that A is the underlying object of an H-algebra $a : HA \longrightarrow A$. Then a *solution* of e in A is a morphism $e^\dagger : X \longrightarrow A$ such that the following diagram

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ HX + A & \xrightarrow[He^\dagger + A]{} & HA + A \end{array} \tag{3.27}$$

commutes.

An H-algebra is called *completely iterative* (or shortly, *cia*) if every flat equation morphism in it has a unique solution.

Notation 3.3. For every flat equation morphism $e : X \longrightarrow HX + Y$ and every morphism $h : Y \longrightarrow Z$ we get a flat equation morphism $h \bullet e$ as the "renaming of parameters by h":

$$h \bullet e \equiv X \xrightarrow{\ e\ } HX + Y \xrightarrow{\ HX + h\ } HX + Z\,.$$

Proposition 3.4. ([M$_1$], Proposition 2.3)
Let (A, a) and (B, b) be completely iterative H-algebras, and let $h : A \longrightarrow B$ be a morphism. Then the following are equivalent:

(i) $h : (A, a) \longrightarrow (B, b)$ is an H-algebra homomorphism,

(ii) h is solution-preserving, i. e., for all $e : X \longrightarrow HX + A$ we have

$$(h \bullet e)^\dagger = h \cdot e^\dagger\,.$$

Notation 3.5. We denote by

$$\mathsf{CIA}\, H$$

the category of all completely iterative algebras and H-algebra homomorphisms. This choice of morphisms is explained by Proposition 3.4, and $\mathsf{CIA}\, H$ is a full subcategory of $\mathsf{Alg}\, H$, the category of all H-algebras and homomorphisms.

Examples 3.6.

(i) Classical algebras are seldom cias. For example, let $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$ be the functor expressing one binary operation, $H_\Sigma X = X \times X$. Then a group is a cia if and only if its unique element is the unit 1, since the recursive equation $x \approx x \cdot 1$ has a unique solution. A lattice is a cia if and only if it has a unique element; consider the unique solution of $x \approx x \vee x$.

(ii) In [AMV$_2$] it was proved that the algebra of addition on the augmented positive natural numbers

$$\{\,1, 2, 3, \dots\,\} \cup \{\,\infty\,\}$$

is a cia w. r. t. the functor H_Σ of (i).

(iii) Final coalgebras are completely iterative algebras. More precisely, denote by (T, α) a final coalgebra for H. Recall that by Lambek's Lemma 2.28, the structure map α is an isomorphism, whose inverse we denote by $\tau : HT \longrightarrow T$. Then this H-algebra (T, τ) is completely iterative. In fact, consider a flat equation morphism

$$e : X \longrightarrow HX + T\,,$$

and form the H-coalgebra

$$\overline{e} \equiv X + T \xrightarrow{\ [e, \mathsf{inr}]\ } HX + T \xrightarrow{\ HX + \alpha\ } HX + HT \xrightarrow{\ \mathrm{can}\ } H(X + T)\,.$$

Then the left-hand component of the unique coalgebra homomorphism $[\![\overline{e}]\!] : X + T \longrightarrow T$ is the desired unique solution $e^\dagger$ of e.

(iv) Infinite trees form completely iterative algebras. Let Σ be a signature. Recall from Example 2.29(vi) that the algebras Σ-algebra T_Σ of all (finite and infinite) Σ-trees is a final H_Σ-coalgebra. Thus, T_Σ is a cia.

(v) The final coalgebra for $\mathcal{P}_{\mathrm{fin}} : \mathsf{Set} \longrightarrow \mathsf{Set}$ is the coalgebra T of all strongly extensional finitely branching trees, see Example 2.29(iv). It follows from (iii) that T is a cia.

(vi) Complete metric spaces have been used as a basis for semantics by several authors, see [AN, ARu, ARe, TR]. Algebras over complete metric spaces are cias. Take $\mathcal{A} = \mathsf{CMS}$, the category whose objects are complete metric spaces (i. e., such that each Cauchy sequence has a limit), where distances are measured in the interval $[0, 1]$. The morphisms of CMS are the *non-expanding maps*, i. e., functions f from a space (X, d_X) to a space (Y, d_Y) such that $d_Y(f(x), f(y)) \leq d_X(x, y)$ for all $x, y \in X$. A stronger condition is that f be an *ε-contraction*, i. e., for some $\varepsilon < 1$ we have $d_Y(f(x), f(y)) \leq \varepsilon \cdot d_X(x, y)$ for all $x, y \in X$. Recall that for given complete metric spaces (X, d_X) and (Y, d_Y) the hom-set in CMS is a complete metric space with the metric given by

$$d_{X,Y}(f, g) = \sup_{x \in X} d_Y(f(x), g(x)) .$$

Now suppose we have a functor $H : \mathsf{CMS} \longrightarrow \mathsf{CMS}$ which is *contracting*, i. e., there exists a constant $\varepsilon < 1$ such that for any non-expanding maps $f, g : (X, d_X) \longrightarrow (Y, d_Y)$ between complete metric spaces we have

$$d_{HX,HY}(Hf, Hg) \leq \varepsilon \cdot d_{X,Y}(f, g) .$$

Then every non-empty H-algebra (A, a) is completely iterative. In fact, given any flat equation morphism $e : X \longrightarrow HX + A$ in CMS, choose some element $a \in A$ and define a Cauchy sequence $(e^\dagger_n)_{n \in \mathbb{N}}$ in $\mathsf{CMS}(X, A)$ inductively as follows: let $e^\dagger_0 = \mathsf{const}_a$, and given $e^\dagger_n$ define $e^\dagger_{n+1}$ by the commutativity of the following diagram (compare (3.27))

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger_{n+1}} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ HX + A & \xrightarrow[He^\dagger_n + A]{} & HA + A \end{array} \tag{3.28}$$

In [AMV$_3$], Lemma 2.9, it is proved that this is indeed a Cauchy sequence in $\mathsf{CMS}(X, A)$ and that its limit yields a unique solution of e.

(vii) Completely metrizable algebras. Many set functors H have a lifting to contracting endofunctors H' of CMS. That is, for the forgetful functor $U : \mathsf{CMS} \longrightarrow \mathsf{Set}$ the following square

$$\begin{array}{ccc} \mathsf{CMS} & \xrightarrow{H'} & \mathsf{CMS} \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathsf{Set} & \xrightarrow[H]{} & \mathsf{Set} \end{array}$$

commutes. For example, for the functor $HX = X^n$ we have the lifting $H'(X, d) = (X^n, \frac{1}{2} \cdot d_{\max})$, where $d_{\max}$ is the maximum metric, which is a contracting functor with $\varepsilon = \frac{1}{2}$. Since coproducts of $\frac{1}{2}$-contracting liftings are $\frac{1}{2}$-contracting liftings of coproducts, we conclude that every polynomial endofunctor has a contracting lifting to CMS.

Let $\alpha : HA \longrightarrow A$ be an H-algebra such that there exists a complete metric, d, on A such that α is a non-expanding map from $H'(A, d)$ to (A, d). Then A is a completely iterative H-algebra. In fact, to every equation morphism $e : X \longrightarrow HX + A$ assign the unique solution of $e : (X, d_0) \longrightarrow H'(X, d_0) + (A, d)$, where d_0 is the discrete metric ($d_0(x, x') = 1$ if and only if $x \neq x'$).

For example, let A be the interval $[\frac{3}{2}, 2]$ and let α be the unary operation $\alpha(x) = 1 + \frac{1}{x}$. Then (A, α) is an algebra for the functor $H = Id$ on Set, and this algebra is completely metrizable. In fact, α is a $\frac{1}{2}$-contraction of $[\frac{3}{2}, 2]$, thus, (A, α) is an algebra for the lifting $H'(X, d) = (X, \frac{1}{2} \cdot d)$ of H. Now consider the formal equation $x \approx \alpha(x)$ that gives a flat equation morphism $e : 1 \longrightarrow H1 + A$. Its unique solution $e^\dagger : 1 \longrightarrow A$ chooses the golden ratio φ.

(viii) Non-empty compact subsets form cias. Let (X, d) be a complete metric space. Consider the set $C(X)$ of all non-empty compact subspaces of X together with the so-called Hausdorff metric h; for two compact subsets A and B of X the distance $h(A, B)$ is the smallest number r such that B can be covered by open balls of radius r around each point of A, and vice versa, A can be covered by such open balls around each point of B. In symbols, $h(A, B) = \max\{\, d(A \to B), d(B \to A)\,\}$, where $d(A \to B) = \max_{a\in A} \min_{b\in B} d(a, b)$. It is well-known that $(C(X), h)$ forms a complete metric space, see e.g. [Ba]. Furthermore, if $f_i : X \longrightarrow X$, $i = 1, \ldots, n$, are contractions of the space X with contraction factors c_i, $i = 1, \ldots, n$, then it is easy to show that the map

$$\alpha_X : C(X)^n \longrightarrow C(X) \qquad (A_i)_{i=1,\ldots,n} \longmapsto \bigcup_{i=1}^{n} f_i[A_i]$$

is a contraction with contraction factor $c = \max_i c_i$ (the product $C(X)^n$ is, of course, equipped with the maximum metric). In other words, given the f_i, we obtain on $C(X)$ the structure α_X of an H-algebra for the contracting endofunctor $H(X, d) = (X^n, c \cdot d_{\max})$. Thus, whenever X is non-empty, then $(C(X), \alpha_X)$ is a cia.

(ix) The Cantor "middle-third" set c is the unique non-empty compact subset of the interval $[0, 1]$ which satisfies $c = \frac{1}{3}c \cup (\frac{2}{3} + \frac{1}{3}c)$. So let (X, d) be the Euclidean interval $I = [0, 1]$ and consider the $\frac{1}{3}$-contracting functions $f(x) = \frac{1}{3}x$ and $g(x) = \frac{1}{3}x + \frac{2}{3}$ on I. Then $\alpha_I : C(I)^2 \longrightarrow C(I)$ with $\alpha_I(A, B) = f[A] \cup g[B]$ gives the structure of a cia on $C(I)$ for the functor $H(X, d) = (X^2, \frac{1}{3} \cdot d_{\max})$, which is a lifting of the polynomial endofunctor $H_\Sigma X = X \times X$ of Set expressing one binary operation symbol α. Now consider the formal equation

$$x \approx \alpha(x, x)$$

which gives rise to a flat equation morphism $e : 1 \longrightarrow H1 + C(I)$, where 1 denotes the trivial one point space. Its unique solution $e^\dagger : 1 \longrightarrow C(I)$ is easily seen to choose the Cantor space.

(x) Continuing with our last point, for each non-empty compact $t \in C(I)$, there is a unique $s = s(t)$ with $s = \alpha(s, t)$. The argument is just as above. But the work we have done does *not* show that the map $t \longmapsto s(t)$ is continuous. For this, we would have to study a recursive program scheme $\varphi(x) \approx \alpha(\varphi(x), x)$ and solve this in $(C(I), \alpha_I)$ in the appropriate sense. Our work in Section 7 does exactly this, and it follows that the solution to $\varphi(x) \approx \alpha(\varphi(x), x)$ in the given algebra is the *continuous* function $t \longmapsto s(t)$.

(xi) (Unary algebras over Set)
Here we have $\mathcal{A} = \mathsf{Set}$ and $H = Id$. A unary algebra (A, α_A) is completely iterative if and only if

(a) there exists a unique fixed point $a_0 \in A$ of all $\alpha_A^k : A \longrightarrow A$, $k \geq 1$,

(b) for every sequence $(b_i)_{i<\omega}$ in A with $b_i = \alpha_A(b_{i+1})$ we have $b_i = a_0$ for every $i < \omega$ (i. e., if a is an element of A in which an infinite *alpha*-chain of elements of A ends, then $a = a_0$).

To see that (a) and (b) are necessary, solve the equation $x \approx \alpha(x)$ to obtain the fixed point a_0. Furthermore, the system

$$x_i \approx \alpha(x_{i+1}), \qquad i < \omega,$$

has as solutions every sequence as in (b); in particular, the constant sequence at a_0 is a solution, and this must be the unique one.

For the sufficiency, suppose that (A, α_A) satisfies (a) and (b). Given any equation morphism $e : X \longrightarrow H_\Sigma X + A$ there is a unique solution $e^\dagger : X \longrightarrow A$: If $x \in X$ is such that there exist equations

$$\begin{aligned} x = x_0 &\approx \alpha(x_1) \\ x_1 &\approx \alpha(x_2) \\ &\vdots \\ x_{k-1} &\approx \alpha(x_k) \\ x_k &\approx a \end{aligned}$$

where $a \in A$, then $e^\dagger(x_k) = a$ and therefore $e^\dagger(x) = \alpha^k(a)$. Otherwise we have equations

$$\begin{aligned} x = x_0 &\approx \alpha(x_1) \\ x_1 &\approx \alpha(x_2) \\ x_2 &\approx \alpha(x_3) \\ &\vdots \end{aligned}$$

and (a) and (b) ensure that the unique solution is given by $e^\dagger(x_i) = a_0$, for all i.

We have seen above that final coalgebras yield cias. In fact, they are precisely the initial ones.

Theorem 3.7. ([M$_1$], Theorem 2.8)
Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be any endofunctor.

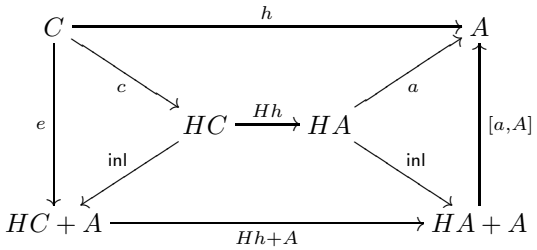
(i) If (T, α) is a final H-coalgebra, then (T, τ) with $\tau = \alpha^{-1}$ is an initial completely iterative H-algebra.

(ii) Conversely, if (T, τ) is an initial completely iterative H-algebra, then τ is an isomorphism and (T, α) with $\alpha = \tau^{-1}$ is a final H-coalgebra.

Sketch of Proof. Before we prove the two statements we shall establish one useful fact about the relation between H-coalgebras and cias. Suppose that (C, c) is any H-coalgebra and (A, a) is a cia. We can form an equation morphism

$$e \equiv C \xrightarrow{c} HC \xrightarrow{\mathsf{inl}} HC + A\,.$$

Then there is a one-to-one correspondence between solutions of e and morphisms $h : C \longrightarrow A$ such that $h = a \cdot Hh \cdot c$ (the so-called coalgebra to algebra homomorphisms). Indeed, this follows easily by inspection of the following diagram:



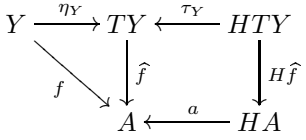
Since there exists a unique solution $e^\dagger$ for e, there exists a unique coalgebra to algebra homomorphism h. It is now quite easy to prove the theorem.

(i) $\Rightarrow$ (ii): We have seen in Example 3.6(iii) that (T, τ) is completely iterative. It remains to prove the initiality. Given any cia (A, a) we have by the above considerations a unique coalgebra to algebra homomorphism $h : T \longrightarrow A$, i. e., unique H-algebra homomorphism $h : (T, \tau) \longrightarrow (A, a)$.

(ii) $\Rightarrow$ (i): One first shows that the structure map of an initial cia (T, τ) is an isomorphism. In fact, $(HT, H\tau)$ is a cia by Proposition 2.6 of [M$_1$]. Thus we obtain an inverse of τ as in the proof of Lambek's Lemma 2.20, and so T is a coalgebra with structure $\alpha = \tau^{-1}$. It remains to show that (T, α) is final. Given any H-coalgebra (C, c) there exists a unique coalgebra to algebra homomorphism $h : C \longrightarrow T$, i. e., a unique H-coalgebra homomorphism $h : (C, c) \longrightarrow (T, \alpha)$. □

Remark 3.8. Notice that in the proof of part "(ii) $\Rightarrow$ (i)" of Theorem 3.7 in lieu of the full universal property of (T, τ) one only uses that the structure map τ is an isomorphism. Thus, the only cia with an isomorphic structure map is the initial one.

By a *free cia* on an object Y of $\mathcal{A}$ we mean, of course, a cia (TY, τ_Y) together with a morphism $\eta_Y : Y \longrightarrow TY$ in $\mathcal{A}$ such that for every cia (A, a) and every morphism $f : Y \longrightarrow A$ in $\mathcal{A}$ there exists a unique homomorphism $\widehat{f} : (TY, \tau_Y) \longrightarrow (A, a)$ extending f, i. e., such that the following diagram



commutes.

The following result shows that free cias correspond to initial cias in a similar way as is the case for ordinary H-algebras.

Theorem 3.9. ([M_1], Theorem 2.10)
For every object Y of $\mathcal{A}$ the following are equivalent:

(i) TY is an initial completely iterative algebra for $H(\_) + Y$.

(ii) TY is a free completely iterative H-algebra on Y.

Corollary 3.10. *Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be any endofunctor.*

(i) If (TY, α_Y) is a final coalgebra for $H(\_) + Y$ the inverse of its structure map gives an H-algebra $\tau_Y : HTY \longrightarrow TY$ and a morphism $\eta_Y : Y \longrightarrow TY$, and these form a free cia on Y.

(ii) Conversely, if (TY, τ_Y) is a free cia on Y via a universal arrow $\eta_Y : Y \longrightarrow TY$ then $[\tau_Y, \eta_Y] : HTY + Y \longrightarrow TY$ is an isomorphism and its inverse yields a final coalgebra for $H(\_) + Y$.

This corollary establishes the desired equivalence of the statements (a) and (b) from the introduction of this section.

Examples 3.11.

(i) The free cias of $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$. Recall from Example 3.6(iv) the algebra T_Σ of all (finite and infinite) Σ-trees. This algebra is a cia. For every set Y the algebra $T_\Sigma Y$ of all Σ-trees over Y, i. e., finite and infinite Σ-trees where, however, leaves can be labelled by constant symbols from Σ or variables from the set Y, is a final coalgebra for $H_\Sigma(\_) + Y$. By Corollary 3.10, this implies that $T_\Sigma Y$ is a free completely iterative Σ-algebra on Y.

(ii) The free cias of $\mathcal{P}_{\mathrm{fin}} : \mathsf{Set} \longrightarrow \mathsf{Set}$. Recall the final coalgebra T of $\mathcal{P}_{\mathrm{fin}}$ from Example 3.6(v). Analogously, for a set Y a final coalgebra for $\mathcal{P}_{\mathrm{fin}}(\_) + Y$ is the algebra $T(Y)$ of all finitely branching strongly extensional trees with leaves partially labelled in the set Y. By Corollary 3.10, this implies that $T(Y)$ is a free cia on Y.

(iii) The free cias of a finitary set endofunctor. Let $H : \mathsf{Set} \longrightarrow \mathsf{Set}$ be a finitary functor. Recall from Example 2.29(vii) that H is a quotient of some polynomial functor H_Σ via $\varepsilon : H_\Sigma \longrightarrow H$ and that the final coalgebra for H is $T_\Sigma/{\sim^*}$. Analogously, for each set Y we obtain $H(\_) + Y$ as a quotient of $H_\Sigma(\_) + Y$ via $\varepsilon' = \varepsilon + id$, see [$AM_1$]. Notice that the basic equations for ε' are essentially the same as those for ε. Thus $TY = T_\Sigma Y/{\sim^*_Y}$, where $\sim^*_Y$ is defined precisely as $\sim^*$ for T_Σ in Example 2.29(vii). For example, for the functor $\mathcal{P}_2$ assigning to a set Y the set of unordered pairs of Y, TY is the coalgebra of all unordered binary trees with leaves labelled in the set Y.

Definition 3.12. We call an endofunctor H *iteratable*, if it has for every object Y of $\mathcal{A}$ a final coalgebra TY of $H(\_) + Y$.

Corollary 3.13. *For an iteratable endofunctor H the assignment $Y \longmapsto TY$ of a free cia to an object yields a monad $\mathbb{T} = (T, \eta, \mu)$.*

In fact, the unit is given by the universal arrows $\eta_Y : Y \longrightarrow TY$, and the monad multiplication $\mu_Y : TTY \longrightarrow TY$ is obtained as the unique homomorphic extension of id_{TY}. In other words, $\mathbb{T}$ is the monad arising from the adjunction

$$\mathsf{CIA}\, H \underset{U}{\overset{\perp}{\rightleftarrows}} \mathcal{A}$$

where the left-adjoint assigns to every object of $\mathcal{A}$ a free cia on that object, see Subsection 2.4.2.

Observe that an immediate corollary of 3.10 is the Substitution Theorem of [AAMV].

Theorem 3.14. ([AAMV], Theorem 2.17)
Let H be an iteratable endofunctor. Then for every substitution $s : X \longrightarrow TY$ there exists a unique H-algebra homomorphism $\widehat{s} : TX \longrightarrow TY$ with $\widehat{s} \cdot \eta_X = s$.

Notice that for a polynomial endofunctor H_Σ of Set, this theorem states that substitution works for all (finite and infinite) Σ-trees in precisely the same way as for finite Σ-trees (or terms), see Example 2.23(i).

Remark 3.15. For every cia (A, a) we obtain a unique homomorphism

$$\widetilde{a} : TA \longrightarrow A$$

extending id_A. It is easy to see that this yields the structure of an Eilenberg-Moore algebra for the monad $\mathbb{T}$. For a polynomial endofunctor H_Σ of Set one can think of $\widetilde{a}$ as "computations of all Σ-trees", analogously as for finite Σ-trees in Example 2.23(i).

3.2 Iterative Algebras

In the previous subsection we have seen completely iterative algebras in which every system of flat equations has a unique solution. In applications it is often desirable to be able to obtain solutions only for systems which are finitary, i. e., with only finitely many variables. In fact, Calvin Elgot introduced and studied first iterative theories, which are algebraic theories with unique solutions of certain finitary recursive equations. And Evelyn Nelson introduced in [N] iterative algebras for a signature Σ to obtain an easier approach to iterative theories. Recall from [N] that a Σ-algebra A is called *iterative* if every system (3.23) where the set X of variables is finite has a unique solution in A. So again, the algebra $A = T_\Sigma$ of all Σ-trees is iterative. But also its subalgebra R_Σ of all rational Σ-trees is iterative, where recall that a Σ-tree is rational, if it has (up to isomorphism) only finitely many subtrees. Evelyn Nelson proved that for every set X the set $R_\Sigma X$ of all rational Σ-trees over X carries a free iterative algebra on X, and that the assignment $X \longmapsto R_\Sigma X$ of a free iterative algebra yields the free iterative theory on Σ. We will now introduce iterative algebras for every finitary endofunctor of an lfp category $\mathcal{A}$. Furthermore, we will show that every object of $\mathcal{A}$ generates a free iterative algebra, and we provide a canonical construction of it. We present in this subsection the first part of the paper [AMV$_2$], which is joint work with Jiří Adámek and Jiří Velebil.

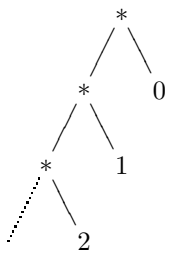
Assumption 3.16. For the rest of this subsection we assume that $\mathcal{A}$ is a locally finitely presentable category and that $H : \mathcal{A} \longrightarrow \mathcal{A}$ is a finitary endofunctor, see Section 2.1

Definition 3.17. A flat equation morphism $e : X \longrightarrow HX + A$ is called *finitary* if X is a finitely presentable object. An H-algebra $a : HA \longrightarrow A$ is called *iterative* if every finitary flat equation morphism e has a unique solution in A, i. e., there exists a unique morphism $e^\dagger : X \longrightarrow A$ such that Diagram (3.27) commutes.

Examples 3.18.

(i) Again, classical algebras are seldom iterative. In fact, notice that in Example 3.6(i) all equations are finitary.

(ii) All completely iterative algebras are obviously iterative. In particular, for a signature Σ, the algebras $T_\Sigma X$ of all Σ-trees over X are iterative H_Σ-algebras. The subalgebras $R_\Sigma X$ of all rational Σ-trees over X are iterative, too, but they usually fail to be completely iterative. For example, let Σ be a signature with a binary operation symbol $*$ and a constant c. Then for rational trees t and s the system (3.26) gives as solutions for x_1 and x_2 the trees in (3.25), which are rational. But the tree



is not rational. And so the system

$$x_i \approx x_{i+1} * i, \qquad i < \omega,$$

gives rise to a flat equation morphism which does not have a solution in $R_\Sigma \omega$.

(iii) The algebra of addition on

$$I = (0, \infty]$$

is iterative, see [AMV$_2$], but it fails to be completely iterative. The equation morphism e given by the system

$$\begin{aligned} x_0 &\approx x_1 + \frac{1}{2} \\ x_1 &\approx x_2 + \frac{1}{4} \\ x_2 &\approx x_3 + \frac{1}{8} \\ &\vdots \end{aligned}$$

has two solutions: one is $e^\dagger(x_n) = \infty$ $(n \in \mathbb{N})$, another one is $e^\dagger(x_n) = 2^{-n}$ $(n \in \mathbb{N})$.

(iv) Unary algebras. An algebra $\alpha : A \longrightarrow A$ of $H = Id$ on Set is iterative if and only if there exists a unique fixed point for each α^n, $n \geq 1$.

Notation 3.19. We denote by $\mathsf{Alg}_{it}\ H$ the full subcategory of $\mathsf{Alg}\, H$ that consists of all iterative H-algebras.

As for cias one proves that the choice of all algebra homomorphisms is appropriate for iterative algebras.

Lemma 3.20. ([AMV$_2$], Lemma 2.15)
Let (A, a) and (B, b) be iterative algebras. Then a morphism f is an H-algebra homomorphism if and only if it preserves solutions of finitary flat equation morphisms, i. e., the equation $(f \bullet e)^\dagger = f \cdot e^\dagger$ holds for all finitary $e : X \longrightarrow HX + A$.

The following result is not difficult to prove. It shows that there are enough iterative algebras to force the existence of free ones.

Proposition 3.21. ([AMV$_2$], Proposition 2.16)
Iterative algebras are closed under limits and filtered colimits.

Corollary 3.22. ([AMV$_2$], Corollaries 2.17 and 2.18)
The category $\mathsf{Alg}_{it}\ H$ is a reflective subcategory of $\mathsf{Alg}\, H$. Thus, every object of $\mathcal{A}$ generates a free iterative H-algebra.

In other words, the natural forgetful functor $U : \mathsf{Alg}_{it}\ H \longrightarrow \mathcal{A}$ has a left adjoint.

Definition 3.23. The finitary monad on $\mathcal{A}$ formed by free iterative H-algebras is called the *rational monad* of H and is denoted by $\mathbb{R} = (R, \eta, \mu)$.

Thus, $\mathbb{R}$ is the monad of the above adjunction

$$\mathsf{Alg}_{it}\ H \underset{U}{\overset{\perp}{\rightleftarrows}} \mathcal{A}$$

More detailed, for every object Y of $\mathcal{A}$ we denote by RY a free iterative H-algebra on Y with the universal arrow

$$\eta_Y : Y \longrightarrow RY\,,$$

and the algebra structure

$$\varrho_Y : HRY \longrightarrow RY\,.$$

Then $\mu_Y : RRY \longrightarrow RY$ is the unique homomorphism of H-algebras with $\mu_Y \cdot \eta_{RY} = id$.

Before turning to concrete examples of free iterative algebras, we will show that it is sufficient to describe the initial one:

Proposition 3.24. ([AMV$_2$], Proposition 2.20)
For every object Y of $\mathcal{A}$ the following are equivalent:

(i) RY is an initial iterative algebra for $H(\_) + Y$,

(ii) RY is a free iterative H-algebra on Y.

In fact, the proof for iterative algebras is the same as for completely iterative algebras, see Theorem 3.9.

Examples 3.25.

(i) The rational monad of $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$. Recall from Example 3.18(ii) that for every set Y the algebra $R_\Sigma Y$ of all rational Σ-trees over Y is iterative. As proved in [N], $R_\Sigma Y$ is a free iterative Σ-algebra on Y. Thus, the rational monad $\mathbb{R}_\Sigma$ of the polynomial endofunctor H_Σ of Set is given by the formation of the Σ-algebras $R_\Sigma Y$ of all rational Σ-trees over Y.

(ii) The rational monad of $\mathcal{P}_{\mathrm{fin}} : \mathsf{Set} \longrightarrow \mathsf{Set}$, the finite power set functor was described in [AM$_1$]: it assigns to a set X the algebra of all rational strongly extensional finitely branching trees; here rational means, again, that there are (up to isomorphism) only finitely many different subtrees.

In Section 2 we have seen that initial algebras and initial cias (equivalently, final coalgebras) can be obtained via a canonical construction. From Corollary 3.22 we know that for every finitary endofunctor on $\mathcal{A}$ an initial iterative algebras exists. We shall now provide a construction of initial iterative algebras. Notice that by Proposition 3.24 this yields a construction of all free iterative algebras.

To motivate our construction recall that for a signature Σ the rational Σ-trees are precisely those Σ-trees which arise as solutions of finitary (flat) systems (3.23) where the right-hand side are terms over X that are not a variable from X.

Now for a finitary functor H on an lfp category $\mathcal{A}$ recall that by $\mathcal{A}_{\mathrm{fp}}$ we denote a chosen set of representatives of all finitely presentable objects of $\mathcal{A}$ w. r. t. isomorphism. Consider the full subcategory EQ of $\mathsf{Coalg}\, H$ formed by all coalgebras $e : X \longrightarrow HX$ with a carrier from $\mathcal{A}_{\mathrm{fp}}$, equivalently, all finitary flat equation morphisms in the initial object of $\mathcal{A}$. We will show that the initial iterative algebra is a colimit of the diagram

$$\mathsf{Eq} : \mathsf{EQ} \longrightarrow \mathcal{A}, \qquad (X, e) \longmapsto X\,.$$

We denote by

$$R_0 = \operatorname{colim} \mathsf{Eq}$$

a colimit of this diagram and we write $e^\sharp : X \longrightarrow R_0$, $e \in \mathsf{EQ}$, for the colimit injections. Notice that we obtain a unique morphism $i : R_0 \longrightarrow HR_0$ so that every $e^\sharp$ becomes a coalgebra homomorphism, i. e., the squares

$$\begin{array}{ccc} X & \xrightarrow{\;e\;} & HX \\ {\scriptstyle e^\sharp}\downarrow & & \downarrow{\scriptstyle He^\sharp} \\ R_0 & \xrightarrow[\;i\;]{} & HR_0 \end{array}$$

commute. In fact, it is easy to check that the morphisms $He^\sharp \cdot e$ form a cocone of Eq.

Theorem 3.26. ([AMV$_2$], Theorem 3.1)
An initial iterative algebra is given by the colimit

$$R_0 = \operatorname{colim} \mathsf{Eq}\,.$$

More precisely, i is an isomorphism whose inverse yields the structure $\varrho_0 : HR_0 \longrightarrow R_0$ of an initial iterative algebra.

Remark. Though easily stated, the proof of this theorem is quite non-trivial. We consider it as one of our main contributions to the subject. The construction of a free iterative algebra is essential for the proofs of most of the subsequent results on the rational monad $\mathbb{R}$ we present below.

Sketch of Proof. (1) We define a morphism

$$j : HR_0 \longrightarrow R_0\,.$$

Notice first that the diagram Eq is filtered. In fact, the category of all coalgebras is cocomplete, with colimits formed at the level of $\mathcal{A}$. Since $\mathcal{A}_{\mathrm{fp}}$ is closed under finite colimits by Proposition 2.5, it follows that the category EQ is closed under finite colimits in the category of all H-coalgebras—thus, EQ is finitely cocomplete, whence filtered.

Consequently, H preserves the colimit of Eq: $HR_0 = \operatorname{colim} H \cdot \mathsf{Eq}$ with the colimit cocone $He^\sharp$. Since $\mathcal{A}$ is locally finitely presentable it follows that HR_0 is a colimit of the diagram D_{HR_0} of all arrows $p : P \longrightarrow HR_0$ where P is in $\mathcal{A}_{\mathrm{fp}}$, see Proposition 2.9. Thus, in order to define j we need to define morphisms $j \cdot p : P \longrightarrow R_0$

forming a cocone of the diagram D_{HR_0}. We know that HR_0 is a filtered colimit of $H \cdot \mathsf{Eq}$ and that P is finitely presentable. Therefore, p factors through one of the colimit morphisms

$$\begin{array}{ccc} P & \xrightarrow{p} & HR_0 \\ & \searrow^{p'} & \uparrow{\scriptstyle Hg^\sharp} \\ & & HW \end{array} \tag{3.29}$$

for some $g : W \longrightarrow HW$ in EQ, see Remark 2.3. We form a new object

$$e_{p'} \equiv P + W \xrightarrow{[p',g]} HW \xrightarrow{H\mathsf{inr}} H(P+W)$$

of EQ and define j to be the unique morphism such that the following square

$$\begin{array}{ccc} P & \xrightarrow{\mathsf{inl}} & P+W \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle e_{p'}{}^\sharp} \\ HR_0 & \xrightarrow[j]{} & R_0 \end{array}$$

commutes for every p in $\mathcal{A}_{\mathrm{fp}}/HR_0$. To prove that j is well-defined ones proves that

(i) $e_{p'}{}^\sharp \cdot \mathsf{inl}$ is independent of the choice of factorization (3.29), and

(ii) the morphisms $e_{p'}{}^\sharp \cdot \mathsf{inl}$ form a cocone of the diagram D_{HR_0}.

And then one readily shows that $j = i^{-1}$.

(2) R_0 is an iterative algebra with structure j. For every equation morphism

$$e : X \longrightarrow HX + R_0 = \mathrm{colim}(HX + \mathsf{Eq})$$

there exists, since X is finitely presentable, a factorization through the colimit morphism $HX + f^\sharp$ (for some $f : V \longrightarrow HV$ in EQ):

$$\begin{array}{ccc} X & \xrightarrow{e} & HX + R_0 \\ & \searrow^{e_0} & \uparrow{\scriptstyle HX+f^\sharp} \\ & & HX + V \end{array}$$

Recall from 3.1 that $\mathsf{can} : HX + HV \longrightarrow H(X+V)$ denotes the canonical morphism. Define a new object, $\overline{e}$, of EQ as follows:

$$\overline{e} \equiv X + V \xrightarrow{[e_0,\mathsf{inr}]} HX + V \xrightarrow{HX+f} HX + HV \xrightarrow{\mathsf{can}} H(X+V)\,.$$

We define a solution of e by

$$e^\dagger \equiv X \xrightarrow{\mathsf{inl}} X + V \xrightarrow{\overline{e}^\sharp} R_0\,.$$

A non-trivial proof now shows that $e^\dagger$ is indeed a solution of e and that it is unique, see [AMV$_2$], Lemma 3.5.

(3) Initiality of the iterative algebra (R_0, j). Let (A, α) be an iterative H-algebra. Consider the equation morphisms

$$X \xrightarrow{e} HX \xrightarrow{\mathsf{inl}} HX + A\,, \qquad e \text{ in } \mathsf{EQ}.$$

Their solutions $(\mathsf{inl} \cdot e)^\dagger : X \longrightarrow A$ form a cocone of Eq. The unique induced morphism $h : R_0 \longrightarrow A$ such that $h \cdot e^\sharp = (\mathsf{inl} \cdot e)^\dagger$ is the desired unique homomorphism of H-algebras (R_0, j) to (A, α). □

Corollary 3.27. ([AMV$_2$], Corollary 3.6)
A free iterative H-algebra RY is a colimit

$$RY = \mathrm{colim}\, \mathsf{Eq}_Y$$

of the diagram

$$\mathsf{Eq}_Y : \mathsf{EQ}_Y \longrightarrow \mathcal{A}\,,$$

where EQ_Y consists of all finitary equation morphisms $e : X \longrightarrow HX + Y$, and all coalgebra homomorphisms w.r.t. $H(-) + Y$, and Eq_Y sends (X, e) to X.

In fact, this is a consequence of Proposition 3.24 and Theorem 3.26.

Remark 3.28. We denote, again, the colimit morphisms of Eq_Y by

$$e^\sharp : X \longrightarrow RY$$

for all $e : X \longrightarrow HX + Y$ in EQ_Y. The appropriate isomorphism is denoted by

$$i_Y : RY \longrightarrow HRY + Y$$

It is characterized by the fact that the two coproduct injections of $HRY + Y$ are (in the notation of Definition 3.23)

$$\mathsf{inl} \equiv HRY \xrightarrow{\varrho_Y} RY \xrightarrow{i_Y} HRY + Y \qquad \text{and} \qquad \mathsf{inr} \equiv Y \xrightarrow{\eta_Y} RY \xrightarrow{i_Y} HRY + Y\,.$$

In other words, $i_Y = [\varrho_Y, \eta_Y]^{-1}$.

Example 3.29. Recall from Example 2.29(iii) that deterministic sequential automata with an input alphabet Σ are the coalgebras for the functor $HX = X^\Sigma \times 2$ on Set, and the final coalgebra T for H is carried by the set of all formal languages over Σ. Since H is a polynomial functor, an initial iterative algebra $R\emptyset$ is carried by a subset of T. For a finite automaton considered as a coalgebra $e : X \longrightarrow HX$ the map $e^\sharp : X \longrightarrow R\emptyset$ assigns to each state the language accepted by X with that state as the initial one. Thus, our construction above explains that $R\emptyset$ is the set of all formal languages accepted by automata with a finite state set, i. e., $R\emptyset$ consists of all regular languages.

4 Elgot Algebras

Unsheathe your dagger definitions.
James Joyce, *Ulysses*

In this section we present the results of our paper [AMV$_3$], which is joint work with Jiří Adámek and Jiří Velebil, again. We study Elgot algebras, a notion of algebra useful for application in the semantics of recursive computations. In fact, in Section 7 we shall use Elgot algebras to study the semantics of recursive program schemes such as (1.7) from the introduction, where two functions are recursively defined from the givens F and G. Recall that one has to distinguish between *uninterpreted* and *interpreted* semantics. In the uninterpreted semantics the givens are not functions but merely function symbols from a signature Σ. In the present section we prepare a basis for the interpreted semantics in which a program scheme comes together with a suitable Σ-algebra A, which gives an interpretation to all the given function symbols. From our discussion in the introduction we have seen that by a suitable Σ-algebra we should understand one in which all Σ-trees, or all rational ones, respectively, have a canonical computation. That means we want to describe precisely those algebras A with a canonical map $T_\Sigma A \longrightarrow A$, or $R_\Sigma A \longrightarrow A$.

More generally, we have seen in the previous section that we can abstract away from signatures and sets. For an endofunctor H of a category $\mathcal{A}$ with finite coproducts the final coalgebras TA for $H(\_) + A$ for every object A yield, if they exist, free cias. Analogously, for a finitary endofunctor H of a locally finitely presentable category $\mathcal{A}$ every object A of $\mathcal{A}$ generates a free iterative algebra RA. Our question then is: what is the largest category of H-algebras in which TA, or RA, respectively, act as free algebras on A? The answer in case of TA is: complete Elgot algebras. These are H-algebras with an additional function assigning to each flat equation morphism e a solution $e^\dagger$. Two (surprisingly simple) axioms are put on $(-)^\dagger$ which stem from the structure of TA: the free cias TA yield the monad $\mathbb{T}$, see Corollary 3.13, and complete Elgot algebras form precisely the category of Eilenberg-Moore algebras for $\mathbb{T}$, i. e., they are precisely those algebras with a "canonical" morphism $TA \longrightarrow A$. Moreover, for an object mapping T of $\mathcal{A}$ we shall add another equivalent description to (a)–(c) of the Introduction of Section 3; (a)–(c) are equivalent to

(d) for every object Y, TY is a free complete Elgot algebra on Y.

We also show that (non-complete) Elgot algebras form the Eilenberg-Moore category of the rational monad $\mathbb{R}$, see Definition 3.23. They are defined precisely as complete Elgot algebras, except that only finitary flat equation morphisms are considered. Basic examples of (complete) Elgot algebras include continuous algebras, metrizable algebras, and, of course, all (completely) iterative algebras.

The two axioms of (complete) Elgot algebras are inspired by the axioms of iteration theories of Stephen Bloom and Zoltán Ésik [BÉ]. In fact, we are able to draw a connection to those structures. For a signature Σ with a distinguished constant symbol $\perp$ the rational trees form an iteration theory whose iteration theory algebras are certain Eilenberg–Moore algebras for the rational monad R_Σ satisfying an extra extensionality property. However, not every Elgot algebra needs to satisfy this extra property, see Example 4.25 below.

4.1 Properties of (Completely) Iterative Algebras

Before we define Elgot algebras let us investigate properties of solutions in iterative algebras and cias.

Remark 4.1. We are going to prove two properties of iterative algebras and cias: the Functoriality and the Compositionality for solutions. We will use two "operations" on equation morphisms. One, $\bullet$, is just change of parameter names: given an equation morphism $e : X \longrightarrow HX + Y$ and a morphism $h : Y \longrightarrow Z$ we obtain the equation morphism $h \bullet e : X \longrightarrow HX + Z$, see Notation 3.3. The other operation $\blacksquare$ combines two flat equation morphisms

$$e : X \longrightarrow HX + Y \qquad \text{and} \qquad f : Y \longrightarrow HY + A$$

into the single flat equation morphism $f \blacksquare e : X + Y \longrightarrow H(X + Y) + A$ in a canonical way:

$$f \blacksquare e \equiv X + Y \xrightarrow{[e,\mathsf{inr}]} HX + Y \xrightarrow{HX+f} HX + HY + A \xrightarrow{\mathsf{can}+A} H(X + Y) + A, \tag{4.30}$$

4.2. Functoriality. This states that solutions are invariant under renaming of variables, provided, of course, that the right-hand sides of equations are renamed accordingly. Formally, observe that every flat equation

morphism is a coalgebra for the endofunctor $H(-)+A$. Given two such coalgebras e and f, a renaming of the variables (or *morphism of equations*) is a morphism $h : X \longrightarrow Y$ which forms a coalgebra homomorphism:

$$\begin{array}{ccc} X & \xrightarrow{e} & HX + A \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh+A} \\ Y & \xrightarrow[f]{} & HY + A \end{array} \tag{4.31}$$

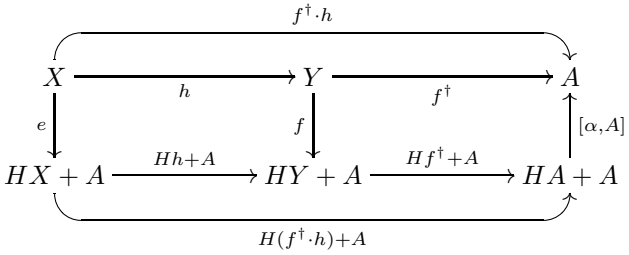
The Functoriality states that

$$e^\dagger = f^\dagger \cdot h \tag{4.32}$$

holds for all equation morphisms h from e to f. In other words: $(-)^\dagger$ is a functor from the category of all flat equation morphisms in the algebra A into the comma-category of the object A.

Lemma 4.3. ([AMV$_3$], Lemma 2.16)
In every cia the assignment $(-)^\dagger$ is functorial.

Proof. For each morphism h of equations the diagram



commutes. Thus, $f^\dagger \cdot h$ is a solution of e. Uniqueness of solutions now implies the desired result. □

Remark 4.4. The same holds for every iterative algebra, except that there we restrict X and Y in 4.2 to finitely presentable objects.

4.5. Compositionality. This tells us how to perform simultaneous recursion: given an equation morphism f in A with a variable object Y, we can combine it with any equation morphism e in Y with a variable object X to obtain the equation morphism $f \blacksquare e$ in A of Remark 4.1. The Compositionality decrees that the left-hand component of $(f \blacksquare e)^\dagger$ is just the solution of $f^\dagger \bullet e$, i. e., in lieu of solving f and e simultaneously we first solve f, plug in the solution in e and solve the resulting equation morphism.

In symbols: given $f : Y \longrightarrow HY + A$ and $e : X \longrightarrow HX + Y$ the Compositionality states that

$$(f^\dagger \bullet e)^\dagger = (f \blacksquare e)^\dagger \cdot \mathsf{inl} \tag{4.33}$$

Remark 4.6. Notice that the coproduct injection $\mathsf{inr} : Y \longrightarrow X + Y$ is a morphism of equations from f to $f \blacksquare e$. The Functoriality then implies that $f^\dagger = (f \blacksquare e)^\dagger \cdot \mathsf{inr}$. Thus, in the presence of Functoriality, the Compositionality is equivalent to

$$(f \blacksquare e)^\dagger = [(f^\dagger \bullet e)^\dagger, f^\dagger]\,. \tag{4.34}$$

Lemma 4.7. ([AMV$_3$], Lemma 2.20)
In every cia the assignment $(-)^\dagger$ satisfies the Compositionality.

Remark 4.8. The same holds for every iterative algebra, except that there we restrict X and Y in 4.5 to finitely presentable objects.

Remark 4.9. As mentioned in the introduction of this section, our two axioms, Functoriality and Compositionality, are not new as ideas of axiomatizing recursion—we believe however, that their concrete form is new, and their motivation strengthened by the results below.

The Functoriality resembles the "functorial dagger implication" of Stephen Bloom and Zoltán Ésik [BÉ], 5.3.3. And the Compositionality resembles the "left pairing identity" of [BÉ], 5.3.1. This identity corresponds also to the Bekić-Scott identity, see e. g. [Mo$_1$], 2.1.

4.2 The Category of Elgot Algebras

Definition 4.10. Let H be an endofunctor of a category with finite coproducts. An *Elgot algebra* is an H-algebra $\alpha : HA \longrightarrow A$ together with a function $(-)^\dagger$ which to every finitary flat equation morphism

$$e : X \longrightarrow HX + A$$

assigns a solution $e^\dagger : X \longrightarrow A$ in such a way that the Functoriality (4.32) and the Compositionality (4.33) are satisfied.

By a *complete Elgot algebra* we analogously understand an H-algebra together with a function $(-)^\dagger$ assigning to every flat equation e a solution $e^\dagger$ so that Functoriality and Compositionality are satisfied.

Examples 4.11.

(i) Every join semilattice A is an Elgot algebra. More precisely: consider the polynomial endofunctor $HX = X \times X$ of Set (expressing one binary operation). Then for every join semilattice A there is a "canonical" structure of an Elgot algebra on A obtained as follows: the algebra RA of all rational binary trees on A has an interpretation on A given by the function $\alpha : RA \longrightarrow A$ forming, for every rational binary tree t the join of all the (finitely many) labels of leaves of t in A. Now given a finitary flat equation morphism $e : X \longrightarrow X \times X + A$, it has a unique solution $e^\dagger : X \longrightarrow RA$ in the free iterative algebra RA, and composed with α this yields a structure $e \longmapsto \alpha \cdot e^\dagger$ of an Elgot algebra on A. See Example 4.24 for a proof. Recall that in contrast, no nontrivial join semilattice is iterative: it has too many idempotents, i. e., solutions of $x \approx x \vee x$, see Example 3.6(i).

(ii) Continuous algebras on cpos are complete Elgot algebras. Let us work here in the category

$$\mathsf{CPO}$$

of all *ω-complete posets*, i.e., posets (not necessarily with a least element) having joins of increasing ω-chains; morphisms are the *continuous functions*, i.e., functions preserving joins of ω-chains. Observe that the category CPO has coproducts: they are the disjoint unions with elements of different summands incompatible.

A functor $H : \mathsf{CPO} \longrightarrow \mathsf{CPO}$ is called *locally continuous* provided that for arbitrary cpos, X and Y, the derived function from $\mathsf{CPO}(X, Y)$ to $\mathsf{CPO}(HX, HY)$ is continuous (i.e., $H(\bigsqcup f_n) = \bigsqcup Hf_n$ holds for all increasing ω-chains $f_n : X \longrightarrow Y$). For example, every polynomial endofunctor $X \longmapsto \coprod_n \Sigma_n \times X^n$ of CPO (where Σ_n are cpos) is locally continuous.

Let $H : \mathsf{CPO} \longrightarrow \mathsf{CPO}$ be a locally continuous functor. Every H-algebra $\alpha : HA \longrightarrow A$ with a least element $\bot \in A$ is a complete Elgot algebra provided that to every equation morphism e the least solution $e^\dagger$ is assigned, see [AMV$_3$], Proposition 3.5. Notice that the least solution of $e : X \longrightarrow HX + A$ refers to the element-wise order of the hom-set $\mathsf{CPO}(X, A)$. We can actually prove a concrete formula for $e^\dagger$ as a join of the ω-chain

$$e^\dagger = \bigsqcup_{n \in \omega} e^\dagger_n$$

of "approximations": $e^\dagger_0$ is the constant function to $\bot$, the least element of A, and given $e^\dagger_n$, then $e^\dagger_{n+1}$ is defined by the commutativity of Diagram (3.28).

(iii) Many set functors H have a lifting to locally continuous endofunctors H' of CPO. That is, for the forgetful functor $U : \mathsf{CPO} \longrightarrow \mathsf{Set}$ the following square

$$\begin{array}{ccc} \mathsf{CPO} & \xrightarrow{H'} & \mathsf{CPO} \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathsf{Set} & \xrightarrow[H]{} & \mathsf{Set} \end{array}$$

commutes. For example, every polynomial functor H_Σ has such a lifting. Let $\alpha : HA \longrightarrow A$ be an H-algebra such that there exists a CPO-ordering $\sqsubseteq$ with a least element on the set A such that α is a continuous function from $H'(A, \sqsubseteq)$ to $(A, \sqsubseteq)$. Then A is a complete Elgot algebra for H. In fact, to every equation morphism $e : X \longrightarrow HX + A$ assign the least solution of $e : (X, \leq) \longrightarrow H'(X, \leq) + (A, \sqsubseteq)$ where $\leq$ is the discrete ordering of X ($x \leq y$ if and only if $x = y$). We shall see in Example 4.20 below that not every complete Elgot algebra needs to arise as a CPO-enrichable one.

(iv) Unary algebras. Let $H = Id$ as an endofunctor of Set. Given an H-algebra $\alpha : A \longrightarrow A$, if α has no fixed point, then A carries no structure of an Elgot algebra: consider the formal equation $x \approx \alpha(x)$.

Conversely, every fixed point a_0 of α yields a flat cpo structure with a least element a_0 on A, i. e., $x \leq y$ if and only if $x = y$ or $x = a_0$. Thus, A is a complete Elgot algebra since it is CPO-enrichable.

(v) Every complete lattice A is a complete Elgot algebra for $HX = X \times X$. Analogously to Example 4.11(i) we have a function $\alpha : TA \longrightarrow A$ assigning to every binary tree t in TA the join of all labels of leaves of t in A. Now for every flat equation morphism e in A we have its unique solution $e^\dagger$ in TA and this yields a structure $e \longmapsto \alpha \cdot e^\dagger$ of a complete Elgot algebra. See Example 4.19 for a proof.

4.3 The Eilenberg-Moore Algebras for the Monad $\mathbb{T}$

Recall our standing assumptions that H is an endofunctor of a category $\mathcal{A}$ with finite coproducts, and recall the operation $\bullet$ and $\blacksquare$ of Remark 4.1.

Definition 4.12. A *homomorphism* h from a complete Elgot algebra $(A, a, (-)^\dagger)$ to a complete Elgot algebra $(B, b, (-)^\ddagger)$ is a morphism $h : A \longrightarrow B$ preserving solutions: for each $e : X \longrightarrow HX + A$ we have the following equation

$$X \xrightarrow{e^\dagger} A \xrightarrow{h} B \equiv X \xrightarrow{(h \bullet e)^\ddagger} B\,. \tag{4.35}$$

Lemma 4.13. ([AMV$_3$], Lemma 5.2)
Every homomorphism $h : (A, a, (-)^\dagger) \longrightarrow (B, b, (-)^\ddagger)$ of complete Elgot algebras is a homomorphism of H-algebras.

Example 4.14. The converse of Lemma 4.13 is true for completely iterative algebras, as proved in Proposition 3.4, but for complete Elgot algebras in general it is false. In fact, consider the unary algebra $id : A \longrightarrow A$, where $A = \{0, 1\}$. This is a complete Elgot algebra with the solution structure $(-)^\dagger$ given by the fixed point $0 \in A$, see Example 4.11(v).

Then $\mathrm{const}_1 : A \longrightarrow A$ is a homomorphism of unary algebras that does not preserve solutions. Indeed, consider the following equation morphism

$$e : \{x\} \longrightarrow \{x\} + A, \quad x \longmapsto x.$$

We have $e^\dagger(x) = 0$, and thus $1 = \mathrm{const}_1 \cdot e^\dagger(x) \neq (\mathrm{const}_1 \bullet e)^\dagger(x) = e^\dagger(x) = 0$.

Notation 4.15. We denote by

$$\mathsf{Alg}_c^\dagger H$$

the the (non-full) subcategory of $\mathsf{Alg}\, H$ formed by all complete Elgot algebras and their homomorphism.

The following result establishes that statement (d) from the introduction of this section is equivalent to (a)–(c) in Section 3, see also Corollary 3.10.

Theorem 4.16. ([AMV$_3$], Theorem 5.3)
Let Y be an object of $\mathcal{A}$. Then the following are equivalent:

(1) TY is a final coalgebra for $H(\_) + Y$, and

(2) TY is a free complete Elgot algebra on Y.

Sketch of Proof. Suppose first that (TY, α_Y) is a final coalgebra for $H(\_) + Y$. Let $[\tau_Y, \eta_Y]$ be the inverse of α_Y. Then $\tau_Y : HTY \longrightarrow TY$ is a completely iterative algebra, see Corollary 3.10, and therefore an Elgot algebra.

Furthermore, $(TY, \tau_Y, (-)^\dagger)$ is a free Elgot algebra on Y. For any given Elgot algebra $(A, a, (-)^\ddagger)$ and any given morphism $m : Y \longrightarrow A$ form the equation morphism

$$m \bullet \alpha_Y \equiv TY \xrightarrow{\alpha_Y} HTY + Y \xrightarrow{HTY + m} HTY + A\,.$$

It is shown in Theorem 5.3 of [AMV$_3$] that the solution $h = (m \bullet \alpha_Y)^\ddagger$ yields the unique homomorphism $h : TY \longrightarrow A$ of Elgot algebras such that $h \cdot \eta_Y = m$.

Now conversely, assume that $(TY, \tau_Y, (-)^\dagger)$ is a free Elgot algebra on Y with a universal arrow $\eta_Y : Y \longrightarrow TY$. It can be shown that $[\tau_Y, \eta_Y]$ is an isomorphism, see [AMV$_3$]. Denote by α_Y its inverse. Then (TY, τ_Y) is a cia; i. e., for every flat equation morphism $e : X \longrightarrow HX + TY$ the solution $e^\dagger$ is unique. In fact, suppose that s is any solution of e. It follows that s is a morphism of equations from e to the flat equation morphism

$$f \equiv TY \xrightarrow{\alpha_Y} HTY + Y \xrightarrow{HTY + \eta_Y} HTY + TY\,.$$

Thus, $f^\dagger \cdot s = e^\dagger$ by the Functoriality of $(-)^\dagger$. Next one can show, using the Compositionality, that $f^\dagger : TY \longrightarrow TY$ is a homomorphism of Elgot algebras satisfying $f^\dagger \cdot \eta_Y = \eta_Y$. Thus, by the freeness of TY, $f^\dagger = id$. This proves that (TY, τ_Y) is a cia, which implies that it is the free one on Y. It is not difficult to show that this yields a final coalgebra for $H(\_) + Y$. In fact, for any coalgebra $c : C \longrightarrow HC + Y$ the unique solution of the flat equation morphism $\eta_Y \bullet c$ yields a unique homomorphism $(C, c) \longrightarrow (TY, \alpha_Y)$ of coalgebras. □

The proof of this Theorem 4.16 is once again non-trivial, see [AMV$_3$] for the details. It is another important contribution and it is the key ingredient needed to prove the following result. Recall that a functor is called iteratable, if the final coalgebra TX for $H(\_) + X$ exists for every object X of $\mathcal{A}$.

Theorem 4.17. ([AMV$_3$], Theorem 5.7)
If H is an iteratable functor, then the category $\mathsf{Alg}_c^\dagger H$ of complete Elgot algebras is isomorphic to the Eilenberg–Moore category $\mathcal{A}^\mathbb{T}$ of monadic $\mathbb{T}$-algebras (for the free cia monad $\mathbb{T}$ of H).

Remark 4.18. By Theorem 4.16, the natural forgetful functor $U : \mathsf{Alg}_c^\dagger H \longrightarrow \mathcal{A}$ has a left adjoint $Y \longmapsto TY$. Thus, the monad obtained from this adjunction is $\mathbb{T}$, see Corollary 3.13. To prove that the comparison functor $\mathsf{Alg}_c^\dagger H \longrightarrow A^\mathbb{T}$ is an isomorphism the easiest proof we know uses Beck's Theorem, see Theorem 2.50. But it is not very intuitive. A slightly more technical (and much more illuminating) proof has the following sketch: Denote for any object Y by $(TY, \tau_Y, (-)^\ddagger)$ a free Elgot algebra on Y with a universal arrow $\eta_Y : Y \longrightarrow TY$.

(i) For every $\mathbb{T}$-algebra $a_0 : TA \longrightarrow A$ we have an "underlying" H-algebra

$$\alpha \equiv HA \xrightarrow{H\eta_A} HTA \xrightarrow{\tau_A} TA \xrightarrow{a_0} A,$$

and the following formula for solving equations: given a flat equation morphism $e : X \longrightarrow HX + A$ put

$$e^\dagger \equiv X \xrightarrow{(\eta_A \bullet e)^\ddagger} TA \xrightarrow{a_0} A\,.$$

It is not difficult to see that this formula indeed yields a choice of solutions satisfying the Functoriality and the Compositionality.

(ii) Conversely, given a complete Elgot algebra $(A, a, (-)^\dagger)$, define $a_0 : TA \longrightarrow A$ as the solution $\alpha_A^\dagger$ of the structure of the final coalgebra $\alpha_A : TA \longrightarrow HTA + A$ considered as a flat equation morphism in A. One readily proves that a_0 satisfies the two axioms of an Eilenberg-Moore algebra.

(iii) It is necessary to prove that the above passages extend to the level of morphisms and they form functors which are mutually inverse.

Example 4.19. Let A be a complete lattice. Recall from Example 4.11(vi) the function $\alpha : TA \longrightarrow A$ assigning to every binary tree t in TA the join of all labels of leaves of t in A. Since joins commute with joins it follows that $\alpha : TA \longrightarrow A$ is the structure of an Eilenberg-Moore algebra on A. Thus, A is a complete Elgot algebra as described in Example 4.11(vi).

Example 4.20. Not every complete Elgot algebra for an endofunctor of Set needs to be CPO-enrichable. To see this consider the polynomial endofunctor $HX = X + X + 1$ expressing two unary operations s and t and a constant $\bot$. Its free cia monad T assigns to every set X the set $TX = \{ s, t \}^* \times X + \{ s, t \}^\infty$, where S^∞ denotes the set of all (finite and infinite) sequences of elements of S and S^* denotes the set of all finite such, respectively.

Now consider the set $C = \{ 0, 1 \}$ equipped with the following algebra structure

$$\begin{array}{rcll} \gamma : TC & \longrightarrow & C & \\ (w, i) & \longmapsto & i, & i = 0, 1,\ w \in \{ s, t \}^* \\ v & \longmapsto & \left\{ \begin{array}{ll} 1 & \text{if } v = up,\ u \in \{ s, t \}^*,\ p = ttt \ldots, \\ 0 & \text{else} \end{array} \right. & \end{array}$$

It is not difficult to check that this is a structure of an Eilenberg-Moore algebra for the monad T. Thus, C is a complete Elgot algebra by Theorem 4.17.

The two formal equations

$$x \approx s(x) \qquad \text{and} \qquad x \approx t(x) \tag{4.36}$$

give rise to two flat equation morphism $e, f : 1 \longrightarrow H1 + C$; e and f map the unique element of 1 to the left-hand and middle component of $H1 = 1 + 1 + 1$, respectively. From the definition of $(-)^\dagger$ on C as in Remark 4.18 we see that $e^\dagger : 1 \longrightarrow C$ and $f^\dagger : 1 \longrightarrow C$ pick 0 and 1 in C, respectively. However, if C were CPO-enrichable both solutions would have to choose the smallest element in the order on C. Thus, there does not exist any order of C such that C is a continuous algebra and the Elgot algebra structure associated to γ above assigns to each flat equation morphism its least solution.

4.4 The Eilenberg-Moore Algebras for the Monad $\mathbb{R}$

Throughout this subsection H denotes a finitary endofunctor of a locally finitely presentable category $\mathcal{A}$.

In the following result the concept of *homomorphism of Elgot algebras* is defined analogously to Definition 4.12: the equation (4.35) is required to hold for *finitary* flat equation morphisms e only. We denote by

$$\mathsf{Alg}^\dagger\, H$$

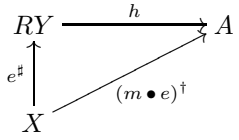
the category of all Elgot algebras and their homomorphisms.

Lemma 4.21. ([AMV$_3$], Lemma 4.2)
Every homomorphism $h : (A, a, (-)^\dagger) \longrightarrow (B, b, (-)^\ddagger)$ of Elgot algebras is a homomorphism of H-algebras.

Notice that although the statement of this result is similar to Lemma 4.13, its proof uses slightly different techniques, see [AMV$_3$]. The converse of Lemma 4.21 is again false, see Example 4.14.

Proposition 4.22. ([AMV$_3$], Proposition 4.6)
A free iterative algebra on Y is a free Elgot algebra on Y.

Sketch of Proof. Since every iterative algebra is an Elgot algebra one only has to prove that a free iterative algebra RY has the universal property. Suppose that $(A, \alpha, (-)^\dagger)$ is an Elgot algebra and let $m : Y \longrightarrow A$ be a morphism. Recall the construction of RY from Theorem 3.26. Define a morphism $h : RY \longrightarrow A$ by commutativity of the following triangles



for all $e : X \longrightarrow HX + Y$ in EQ_Y. In fact, the morphisms $(m \bullet e)^\dagger$ form a cocone of Eq_Y, whence h is well-defined. Now one has to show that h preserves solutions, satisfies the equation $h \cdot \eta_Y = f$, and is uniquely determined, see [AMV$_3$]. □

Theorem 4.23. ([AMV$_3$], Theorem 4.7)
The category $\mathsf{Alg}^\dagger\, H$ of Elgot algebras is isomorphic to the Eilenberg-Moore category $\mathcal{A}^{\mathbb{R}}$ of $\mathbb{R}$-algebras for the rational monad $\mathbb{R}$ of H.

Sketch of Proof. The proof of this theorem is analogous to the proof of Theorem 4.17. By Proposition 4.22 the natural forgetful functor $V : \mathsf{Alg}^\dagger\, H \longrightarrow \mathcal{A}$ has a left adjoint $Y \longmapsto RY$. Thus, the monad obtained by this adjunction is the rational monad $\mathbb{R}$. Ones proves that the comparison functor $\mathsf{Alg}^\dagger\, H \longrightarrow \mathcal{A}^{\mathbb{R}}$ is an isomorphism, using Beck's theorem, see Theorem 2.50. □

Example 4.24. Let A be a join semilattice. Recall from Example 4.11(i) the function $\alpha : RA \longrightarrow A$ assigning to a rational binary tree t in RA the join of the labels of all leaves of t in A. Since joins commute with joins it follows that this is the structure of an Eilenberg-Moore algebra on A. Thus, A is an Elgot algebra as described in Example 4.11(i).

Example 4.25. Iteration algebras of [BÉ] are Elgot algebras. More precisely, let Σ be a signature with a special chosen constant symbol $\bot$. Then the algebraic theory R_Σ formed by all rational Σ-trees is an iteration theory, see [BÉ], Example 6.4.8. Recall that any morphism $n \longrightarrow m$ in this algebraic theory is essentially just a morphism $n \longrightarrow R_\Sigma m$ in the Kleisli category of the monad (associated to) R_Σ, where n and m are considered as sets in the usual way. Furthermore, every interpretation $i : n \longrightarrow A$ of variables in an Eilenberg-Moore algebra A of the monad R_Σ gives rise to a unique homomorphism $\widehat{i} : R_\Sigma n \longrightarrow A$ extending i. Now an R_Σ-iteration algebra in the sense of [BÉ], Definition 7.1.1, is an Eilenberg-Moore algebra (A, α) of the monad R_Σ that satisfies the following extensionality property: Suppose that $f, g : n \longrightarrow n + m$ are two morphisms in the algebraic theory R_Σ, i. e., morphisms $f, g : n \longrightarrow R_\Sigma(n + m)$ in the Kleisli category of the monad R_Σ. If for all interpretations $i : n + m \longrightarrow A$ we have a commuting square

$$\begin{array}{ccc} n+m & \xrightarrow{f} & R_\Sigma(n+m) \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle \widehat{i}} \\ R_\Sigma(n+m) & \xrightarrow[\widehat{i}]{} & A \end{array}$$

then also for all $j : m \longrightarrow A$ the square

$$\begin{array}{ccc} n & \xrightarrow{f^\dagger} & R_\Sigma m \\ {\scriptstyle g^\dagger}\downarrow & & \downarrow{\scriptstyle \widehat{j}} \\ R_\Sigma m & \xrightarrow[\widehat{j}]{} & A \end{array}$$

commutes, where $f^\dagger$ and $g^\dagger$ are the solutions of f and g, respectively, in the iteration theory R_Σ.

Obviously, any R_Σ-iteration algebra is an Elgot algebra for the polynomial functor associated to the signature Σ. But not conversely. In fact, for the signature Σ with two unary operation symbols s and t and the constant symbol $\bot$, the algebra (C, γ) of Example 4.20 yields by precomposition with the inclusion $R_\Sigma C \hookrightarrow T_\Sigma C$ an Eilenberg-Moore algebra for R_Σ, whence an Elgot algebra for H_Σ. However, the two formal equations (4.36) give rise to morphisms $1 \longrightarrow R_\Sigma(1 + 0)$ that violate the desired extensionality property. In fact, we have $\widehat{i} \cdot f = \widehat{i} \cdot g = i$ for all $i : 1 + 0 \longrightarrow C$. But $\widehat{j} \cdot f^\dagger$ picks 0 while $\widehat{j} \cdot g^\dagger$ picks 1 for the unique $j : 0 \longrightarrow C$.

5 Solution Theorems

If you are not part of the solution,
you are part of the problem.
Eldridge Cleaver, *speech in San Francisco*, 1968

In Section 3.1 and 3.2 we started by considering non-flat systems (3.23) of formal recursive equations for Σ-algebras. We then argued that due to the possibility of flattening such a system it suffices to consider flat equation morphisms $X \longrightarrow HX + A$ in general. We shall make this more precise in this section by showing that for all (completely) iterative algebras much more general systems of recursive equations have unique solutions. For polynomial endofunctors of Set, this implies that our notion of iterative algebra coincides with that presented by Evelyn Nelson [N]. As we shall see in the Section 6, this also implies that the rational monad $\mathbb{R}$ is iterative and that the free cia monad $\mathbb{T}$ is completely iterative.

Notice first that the requirement stated in (3.23) that no right-hand side of a system is a variable is important, the equation $x \approx x$ has a unique solution only in the trivial one-point algebra. Systems satisfying the above condition are called *guarded*.

In the current section we present the result of Sections 3 and 4 of [M_1] and [AMV_2], respectively. We start with completely iterative algebras.

5.1 Solution Theorem for Completely Iterative Algebras

We assume in this subsection that $\mathcal{A}$ is a category with finite coproducts. Recall from Corollary 3.13 that for an iteratable endofunctor H, free cias exist on every object Y of $\mathcal{A}$. As before we denote by

$$\tau_Y : HTY \longrightarrow TY \qquad \text{and} \qquad \eta_Y : Y \longrightarrow TY$$

the structure and universal arrow of the free cia TY. Moreover, recall from Remark 3.15 that for every cia (A, a) we denote by

$$\widetilde{a} : TA \longrightarrow A$$

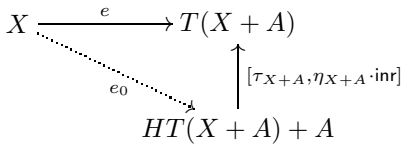
the unique H-algebra homomorphism with $\widetilde{a} \cdot \eta_A = id$.

Definition 5.1. By an *equation morphism* is meant a morphism

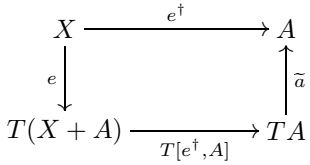
$$e : X \longrightarrow T(X + A)$$

in $\mathcal{A}$ where X is any object (of "variables") and A is any object (of "parameters").

The equation morphism e is called *guarded* if it factors through the summand $HT(X + A) + A$ of $HT(X + A) + X + A = T(X + A)$ (see Corollary 3.10(ii)):



Given a cia (A, a) and an equation morphism e we call a morphism $e^\dagger : X \longrightarrow A$ a solution of e in A if the square



(5.37)

commutes.

Remark 5.2. For a polynomial endofunctor H_Σ of Set an equation morphism e is the same as a system (3.23) where all right-hand sides t_i are Σ-trees over $X + A$, and guardedness of e corresponds precisely to the requirement that no t_i be a variable from X. The commutativity of (5.37) means that the assignment $e^\dagger$ of variables of X to elements of A has the following property: form first the "substitution" mapping $[e^\dagger, A] : X + A \longrightarrow A$ (which replaces variables by their solution according to $e^\dagger$ and leaves parameters

unchanged). Apply this substitution to the right-hand sides of the given system e of formal equations, and compute the resulting infinite trees in A. This yields the same assignment of variables to elements of A as $e^\dagger$. That means that the formal equations $x \approx e(x)$ become actual identities in A after the substitution $x \longmapsto e^\dagger(x)$ is performed on both sides of the equations and the right-hand side is evaluated in A.

Formally, one extends $[e^\dagger, A]$ to the unique homomorphism

$$\widetilde{a} \cdot T[e^\dagger, A] : T(X + A) \longrightarrow A$$

from the free cia on $X + A$ to A. Precomposed with e it yields the morphism $e^\dagger$.

Theorem 5.3. *In every cia, for every guarded equation morphism there exists a unique solution.*

Sketch of Proof. Let (A, a) be a cia. Given a guarded equation morphism e with a factor e_0 as in Definition 5.1, form the flat equation morphism

$$\overline{e} \equiv T(X + A) \xrightarrow{[\tau,\eta]^{-1}} HT(X + A) + X + A \xrightarrow{[\mathsf{inl}, e_0, \mathsf{inr}]} HT(X + A) + A$$

and consider its unique solution $s : T(X + A) \longrightarrow A$ in the cia A. One readily proves that the morphism

$$e^\dagger \equiv X \xrightarrow{\mathsf{inl}} X + A \xrightarrow{\eta} T(X + A) \xrightarrow{s} A$$

is the desired unique solution of e in A. □

Remark 5.4. We have chosen to present in this summary a slightly different version of the solution theorem from [M$_1$]. However, the above Theorem 5.3 follows immediately from Theorem 3.9 of [M$_1$].

5.2 Solution Theorem for Iterative Algebras

It is well-known that for an iterative algebra for a polynomial functor of Set one can uniquely solve guarded system (3.23) where the set of variables is finite and where all right-hand sides t_i are rational trees. We shall now present the corresponding result in our more general setting. We assume in this subsection that $\mathcal{A}$ is an lfp category and that H is a finitary endofunctor of $\mathcal{A}$.

Definition 5.5. By a *rational equation morphism* in an object A we mean a morphism

$$e : X \longrightarrow R(X + A), \qquad X \text{ finitely presentable},$$

and e is called *guarded* if it factors through the summand $HR(X + A) + A$ of $R(X+A) = HR(X+A)+X+A$ (see Remark 3.28):

$$\begin{array}{ccc} X & \xrightarrow{\quad e \quad} & R(X + A) \\ & \searrow^{e_0} & \uparrow{\scriptstyle [\varrho, \eta \cdot \mathsf{inr}]} \\ & & HR(X + A) + A \end{array}$$

Suppose that A is an underlying object of an iterative H-algebra $a : HA \longrightarrow A$. We denote (analogously to $\widetilde{a}$ in the previous subsection) by

$$\widehat{a} : RA \longrightarrow A$$

the unique homomorphism of H-algebras with $\widehat{a} \cdot \eta_A = id$. Then by a *solution* of e in the iterative algebra A is meant a morphism $e^\dagger : X \longrightarrow A$ in $\mathcal{A}$ such that the square

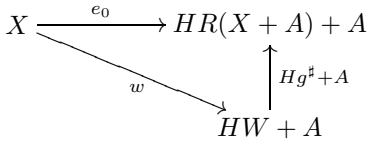
$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \widehat{a}} \\ R(X + A) & \xrightarrow[R[e^\dagger, A]]{} & RA \end{array}$$

commutes.

Theorem 5.6. ([AMV$_2$], Theorem 4.6)
If A is an iterative H-algebra, then every guarded rational equation morphism e in A has a unique solution.

Sketch of Proof. Let $a : HA \longrightarrow A$ be an iterative algebra and let e be a guarded rational equation morphism with a factor e_0 as in Definition 5.5.

Recall from Corollary 3.27 that $R(X+A) = \operatorname{colim} \mathsf{Eq}_{X+A}$ with colimit cocone $g^\sharp : W \longrightarrow R(X+A)$ for all $g : W \longrightarrow HW + X + A$ in EQ_{X+A}. Since this colimit is filtered and H is finitary, we have a filtered colimit $HR(X+A) + A = \operatorname{colim} H\mathsf{Eq}_{X+A} + A$ with the colimit cocone formed by all $Hg^\sharp + A$. Since X is a finitely presentable object, the morphism $e_0 : X \longrightarrow \operatorname{colim} H\mathsf{Eq}_{X+A} + A$ factors through the colimit cocone:



for some object $g : W \longrightarrow HW + X + A$ of EQ_{X+A} and some morphism w.

We define a finitary flat equation morphism as follows:

$$\overline{e} \equiv W + X \xrightarrow{[g,\mathsf{inm}]} HW + X + A \xrightarrow{[\mathsf{inl},w,\mathsf{inr}]} HW + A \xrightarrow{H\mathsf{inl}+A} H(W+X) + A$$

where $\mathsf{inm} : X \longrightarrow HW + X + A$ is the middle coproduct injection. Now consider the unique solution $\overline{e}^\dagger : W + X \longrightarrow A$ in the iterative algebra A. A non-trivial proof shows that the morphism

$$e^\dagger \equiv X \xrightarrow{\mathsf{inr}} W + X \xrightarrow{\overline{e}^\dagger} A$$

is the desired unique solution of e. □

Remark 5.7. It follows from Theorem 5.6 that for a polynomial endofunctor H_Σ of Set our notion of iterative algebra coincides with that of Evelyn Nelson [N]. In fact, she required that iterative algebras have unique solutions of guarded systems (3.23) with finitely many variables where all right-hand sides t_i are terms over $X + A$, i. e. elements of a free H_Σ-algebra on $X + A$. It is not difficult to formulate this for an arbitrary finitary functor on our lfp category $\mathcal{A}$. Indeed, recall that every object X of $\mathcal{A}$ generates a free algebra FX, see Theorem 2.25. Now one formulates the notions of *finitary* equation morphism, solution and guardedness by replacing R by F in Definition 5.5. Then Theorem 5.6 implies that an H-algebra is iterative if and only if it admits unique solutions of all guarded finitary equation morphisms. For details, see [AMV_2], Definition 4.2 and Theorem 4.4.

6 Iterative and Completely Iterative Monads

Und diese Monaden sind die wahrhaften Atomi der Natur
und mit einem Worte / die Elemente derer Dinge.
Gottfried Wilhelm Leibniz, *Monadologie*, 1724

For a polynomial endofunctor of Set Evelyn Nelson [N] has proved that the algebraic theory induced by free iterative algebras is a free iterative theory in the sense of Calvin Elgot [E]. More generally, in this section we will show that the rational monad $\mathbb{R}$ of a finitary endofunctor H, introduced in Definition 3.23, is iterative and it can be characterized as the free iterative monad on H. Similarly, for an iteratable endofunctor H, i. e., all free cias TY of H exist, the free cia monad $\mathbb{T}$, see Corollary 3.13, is a free completely iterative monad on H.

As we shall see the universal property of the monads $\mathbb{R}$ and $\mathbb{T}$ yields an abstract version of the notion of second-order substitution, see Example 2.44. In fact, for a polynomial endofunctor H_Σ of Set the freeness of $\mathbb{T}$ yields second-order substitution of all Σ-trees, and the freeness of $\mathbb{R}$ means that rational trees are closed under this second-order substitution. The (generalized) second-order substitution is a key ingredient of our treatment of the semantics of recursive program schemes in Section 7.

The current section presents the main results of the two papers [AMV$_2$] and [M$_1$].

6.1 Ideal Monads

For a polynomial endofunctor of Set induced by a signature Σ it follows from the universal property of the free completely iterative algebras $T_\Sigma X$ that substitution of trees for variables works for all Σ-trees in precisely the same way as for terms (i. e., all finite Σ-trees), and it follows from the universal property of free iterative algebras $R_\Sigma X$ that rational Σ-trees are closed under substitution. For example, let X be a set of variables and let $s : X \longrightarrow T_\Sigma Y$ be a mapping that assigns to each variable its substitute, a Σ-tree over Y. The unique induced homomorphism $\widehat{s} : T_\Sigma X \longrightarrow T_\Sigma Y$ from the free cia $T_\Sigma X$ on X to $T_\Sigma Y$ with $\widehat{s} \cdot \eta_Y = s$ performs on any given tree of $T_\Sigma X$ the substitution of variables according to s to obtain a tree of $T_\Sigma Y$. Analogously, every substitution $s : X \longrightarrow R_\Sigma Y$ yields by the freeness of the iterative algebra $R_\Sigma X$ a unique H-algebra homomorphism $R_\Sigma X \longrightarrow R_\Sigma Y$ extending s and performing substitution of rational trees.

Notice that the fact that each $\widehat{s} : T_\Sigma X \longrightarrow T_\Sigma Y$ is an H-algebra homomorphism results in the following property of substitution of all Σ-trees: for each tree t which is not just a leaf labelled by a variable, i. e., for all elements of the left-hand coproduct component $H_\Sigma T_\Sigma X$ of $T_\Sigma X$, the result of any substitution will never be just a leaf labelled in Y, i. e., $\widehat{s}(t)$ lies in $H_\Sigma T_\Sigma Y$. Or, more concisely, non-variables are preserved by substitution.

Whereas the concept of variables and substitution is appropriately captured categorically by the concept of a monad, the idea of "non-variable" and its preservation by substitution is not. However, we will need such a concept when we speak of guarded systems of equations for an arbitrary monad below. In fact, in the setting of algebraic theories (i. e., finitary monads on Set) Calvin Elgot [E] introduced the concept of an ideal theory. In [AAMV] we proved that Elgot's ideal theories are equivalent to the following concept.

For a monad $\mathbb{S} = (S, \eta, \mu)$ over Set we can form the complements of the image $\eta_X[X]$ of X under η_X in SX, say,

$$\sigma_X : S'X \longrightarrow SX$$

for all objects X.

The monad is called *ideal* provided $\sigma : S' \longrightarrow S$ is a subfunctor of S, and the monad multiplication has a domain-codomain restriction $\mu' : S'S \longrightarrow S'$. In this subsection we present the corresponding concept for general base categories.

Assumption 6.1. For the rest of Section 6 we assume that $\mathcal{A}$ is a category with finite coproducts such that coproduct injections are monomorphic.

The last requirement that coproduct injections are monomorphic is a mere technicality and could even be totally avoided, see Section 5 of [M$_1$] or Remarks 5.8 and 5.12(1) of [AMV$_2$]. However, we obtain this slightly increased generality at the expense of having to make the following notions and results more technically involved. Hence, we decided to keep the additional requirement for the convenience of the reader and as this property is very common indeed.

Definition 6.2. By an *ideal monad* is understood a six-tuple

$$\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$$

consisting of a monad (S, η, μ), a subfunctor $\sigma : S' \hookrightarrow S$ and a natural transformation $\mu' : S'S \longrightarrow S'$ such that

(i) $S = S' + Id$ with coproduct injections σ and η, and

(ii) μ restricts to μ' along σ, i.e., the following square

$$\begin{array}{ccc} S'S & \xrightarrow{\mu'} & S' \\ {\scriptstyle \sigma S}\downarrow & & \downarrow{\scriptstyle \sigma} \\ SS & \xrightarrow[\mu]{} & S \end{array}$$

commutes.

Examples 6.3.

(i) Free monads are ideal. If $\mathbb{F} = (F, \eta, \mu)$ is a free monad on the endofunctor H with the universal arrow $\kappa : H \longrightarrow F$, then the functor $HF + Id$ is a monad with unit $\widetilde{\eta} = \mathsf{inr} : Id \longrightarrow HF + Id$ and multiplication

$$\begin{array}{rl} \widetilde{\mu} \equiv & (HF + Id)^2 = HF(HF + Id) + HF + Id \\ & \downarrow{\scriptstyle HF(\kappa F + Id) + HF + Id} \\ & HF(FF + Id) + HF + Id \\ & \downarrow{\scriptstyle HF[\mu, \eta] + HF + Id} \\ & HFF + HF + Id \\ & \downarrow{\scriptstyle [H\mu, \mathsf{inl}] + Id} \\ & HF + Id \end{array}$$

see Lemma 3.4 of [M_2]. Thus, the natural transformation $\mathsf{inl} \cdot H\eta : H \longrightarrow HF + Id$ induces a unique monad morphism $\alpha : F \longrightarrow HF + Id$ with $\alpha \cdot \kappa = \mathsf{inl} \cdot H\eta$. It is not difficult to show that the natural transformation

$$\beta \equiv \; HF + Id \xrightarrow{[\kappa F, Id]} FF + Id \xrightarrow{[\mu, \eta]} F$$

is a monad morphism and furthermore an inverse of α. In fact, use the ideas from the proof of Theorem 3.1 of [M_2]. Thus F is a coproduct of HF and Id via the injections $\varphi = \beta \cdot \mathsf{inl}$ and $\eta = \beta \cdot \mathsf{inr}$. Since coproduct injections are monomorphic we obtain a subfunctor $\varphi : HF \hookrightarrow F$ making F an ideal monad. In fact, it is easy to show using the associativity of μ and naturality of κ that the desired restriction of μ is $\mu' = H\mu : HFF \longrightarrow HF$. We leave the details to the interested reader.

(ii) The free cia monad $\mathbb{T} = (T, \eta, \mu)$ together with the endofunctor HT and the natural transformation

$$\tau : HT \hookrightarrow T$$

expressing the H-algebra structure $\tau_Y : HTY \longrightarrow TY$ of each TY is ideal. The restriction of μ here is

$$\mu' = H\mu : HTT \longrightarrow HT\,.$$

In fact, we know that $TY = HTY + Y$ with the coproduct injections τ_Y and η_Y: this follows from Corollary 3.10(ii). And the following square

$$\begin{array}{ccc} HTT & \xrightarrow{H\mu} & HT \\ {\scriptstyle \tau}\downarrow & & \downarrow{\scriptstyle \tau} \\ TT & \xrightarrow[\mu]{} & T \end{array}$$

commutes because each μ_Y is a homomorphism of H-algebras.

(iii) Similarly, if H is a finitary endofunctor of an lfp category $\mathcal{A}$, then the rational monad $\mathbb{R}$ is ideal. In fact, recall from Remark 3.28 that $R = HR + Id$. We consider the subfunctor $HR \hookrightarrow R$ expressing the H-algebra structure $\varrho_Z : HRZ \longrightarrow RZ$ of each RZ. The "restriction" of μ is $\mu' = H\mu$ again.

(iv) The monad on Set given by the free algebras with a binary commutative operation is ideal. In fact, this is the free monad on the endofunctor $\mathcal{P}_2$ that assigns to every set X the set of unordered pairs from X, see Example 2.23(iii) and Theorem 2.42(i).

(v) The free semigroup monad $X \longmapsto X^+$ on Set is ideal. Here X^+ denotes the set of all non-empty words on X and $S'X \hookrightarrow X^+$ is the subset of words of length at least 2, and μ' is the obvious restriction of the concatenation of words to that subset.

(vi) The free monoid monad $X \longmapsto X^*$ on Set is not ideal. In fact, recall that the unit η_X maps elements of X to words of length 1. Now consider the word xx' in $\{ x, x' \}^*$ and the substitution s that substitutes x by itself and x' by the empty word. Then $\widehat{s}(xx') = x$ whence μ cannot have the necessary restriction.

(vii) Classical algebraic theories (groups, lattices, etc.) are usually not ideal.

(viii) Coproducts of ideal monads exist and are ideal. Assume that $\mathcal{A}$ has colimits of ω-chains and let $S = S' + Id$ and $M = M' + Id$ be ideal monads so that S' and M' preserve colimits of ω-chains. Then a coproduct of S and M in the category of monads of $\mathcal{A}$ exists and is an ideal monad, see [GUs].

Definition 6.4.

(i) An *ideal monad morphism* from an ideal monad $(S, \eta^S, \mu^S, S', \sigma, \mu'^S)$ to an ideal monad $(U, \eta^U, \mu^U, U', \omega, \mu'^U)$ is a monad morphism $m : (S, \eta^S, \mu^S) \longrightarrow (U, \eta^U, \mu^U)$ which has a domain-codomain restriction to the ideals. That means that there exists a natural transformation $m' : S' \longrightarrow U'$ such that the square

$$\begin{array}{ccc} S' & \xrightarrow{m'} & U' \\ {\scriptstyle\sigma}\downarrow & & \downarrow{\scriptstyle\omega} \\ S & \xrightarrow[m]{} & U \end{array}$$

commutes.

(ii) Given a functor H, a natural transformation $\lambda : H \longrightarrow S$ is called *ideal* provided that it factors through $\sigma : S' \hookrightarrow S$, i. e., there exists a natural transformation $\lambda' : H \longrightarrow S'$ with $\sigma \cdot \lambda' = \lambda$.

Example 6.5. For every endofunctor H, a free monad $\mathbb{F}$, a rational monad $\mathbb{R}$ and a free cia monad $\mathbb{T}$ (if they exist) come with canonical ideal natural transformations

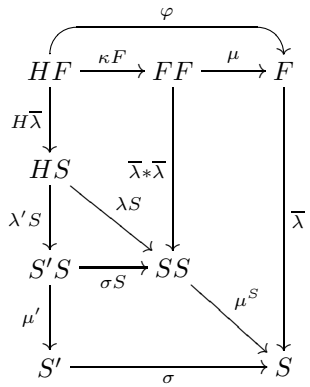
$$H \xrightarrow{H\eta} HF \xrightarrow{\varphi} F\,, \qquad HR \xrightarrow{H\eta} HR \xrightarrow{\varrho} R\,, \qquad HT \xrightarrow{H\eta} HT \xrightarrow{\tau} T\,. \tag{6.38}$$

For a polynomial endofunctor H_Σ of Set these transformations express the operation symbols of the signature Σ (regarded as flat Σ-trees over the set X) as terms in $F_\Sigma X$ or (rational) Σ-trees in $T_\Sigma X$ and $R_\Sigma X$, respectively.

Theorem 6.6. *A free monad $\mathbb{F}$ on H (if it exists) is a free ideal monad on $\mathcal{A}$. More precisely, for any monad $\mathbb{S}$ and any natural transformation $\lambda : H \longrightarrow S$ there exists a unique monad morphism $\overline{\lambda} : \mathbb{F} \longrightarrow \mathbb{S}$ such that $\overline{\lambda} \cdot \kappa = \lambda$. Furthermore, if $\mathbb{S}$ is an ideal monad and λ an ideal natural transformation, then $\overline{\lambda}$ is an ideal monad morphism.*

Proof. If $\mathbb{F} = (F, \eta, \mu)$ is a free monad on H we know from Example 6.3(i) that $\mathbb{F}$ is an ideal monad. We only need to show that the last part of the statement of our Theorem. So consider any ideal monad $\mathbb{S}$ and any ideal natural transformation λ. Then for the unique induced monad morphism $\overline{\lambda} : \mathbb{F} \longrightarrow \mathbb{S}$ consider the

diagram



All its inner parts commute: the upper part is the definition of φ, see Example 6.3, the two parts below it commute since $\overline{\lambda} \cdot \kappa = \lambda$ and since $\overline{\lambda}$ is a monad morphism, the inner triangle commutes since λ is an ideal natural transformation and the lowest part commutes since $\mathbb{S}$ is an ideal monad. Thus, the left-hand edge $\mu' \cdot \lambda' S \cdot H\overline{\lambda} : HF \longrightarrow S'$ is the desired restriction of $\overline{\lambda}$. □

6.2 The Universal Properties

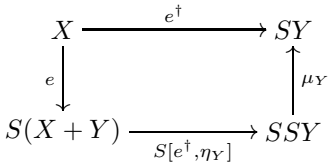
Definition 6.7. Let $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ be an ideal monad on $\mathcal{A}$.

(i) By an *equation morphism* is meant a morphism

$$e : X \longrightarrow S(X + A)$$

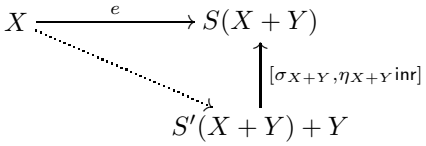
in $\mathcal{A}$ where X is an object ("of variables") and A is an object ("of parameters"). We call e *finitary* if X is a finitely presentable object of $\mathcal{A}$.

(ii) By a *solution* of e is meant a morphism $e^\dagger : X \longrightarrow SA$ for which the square



commutes.

(iii) The equation morphism e is called *guarded* if it factors through the summand $S'(X + Y) + Y$ of $S(X + Y) = S'(X + Y) + X + Y$:



(iv) The ideal monad $\mathbb{S}$ is called *completely iterative* provided that every guarded equation morphism has a unique solution. And $\mathbb{S}$ is called *iterative* if every guarded finitary equation morphism has a unique solution.

Clearly, any completely iterative monad is also iterative. Furthermore, we have the following result:

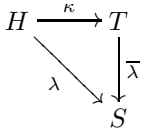
Theorem 6.8.

(i) If H is an iteratable endofunctor, then its free cia monad $\mathbb{T}$ is completely iterative.

(ii) If H is a finitary endofunctor of the lfp category $\mathcal{A}$, then the rational monad $\mathbb{R}$ of H is iterative.

In fact, this follows easily from Theorems 5.3 and 5.6. For the full proof of (ii) see Corollary 4.7 of [AMV_2]. The proof of (i) is completely analogous.

Theorem 6.9. ([M_1], Theorem 4.3)
Let H be an iteratable endofunctor of $\mathcal{A}$. Then the monad $\mathbb{T}$ is a free completely iterative monad on H. That is, the canonical transformation $\kappa = \tau \cdot H\eta : H \longrightarrow T$, see Example 6.5, has the following universal property: for every completely iterative monad $\mathbb{S}$ and every ideal natural transformation $\lambda : H \longrightarrow S$ there exists a unique monad morphism $\overline{\lambda} : \mathbb{T} \longrightarrow \mathbb{S}$ such that the triangle



commutes. Furthermore, this $\overline{\lambda}$ is an ideal monad morphism.

Sketch of Proof. For every object A of $\mathcal{A}$ consider SA as an H-algebra with the structure

$$HSA \xrightarrow{\lambda_{SA}} SSA \xrightarrow{\mu_A} SA\,.$$

This is a completely iterative algebra. In fact, every flat equation morphism $e : X \longrightarrow HX + SA$ yields the following equation morphism w. r. t. $\mathbb{S}$:

$$\overline{e} \equiv X \xrightarrow{\quad e \quad} HX + SA \xrightarrow{\lambda_X + SA} SX + SA \xrightarrow{\quad \mathsf{can} \quad} S(X + A)\,.$$

It is easy to verify that $\overline{e}$ is guarded, and that solutions of $\overline{e}$ w. r. t. the completely iterative monad $\mathbb{S}$ are in 1-1-correspondence with solutions of e. Thus, since $\overline{e}$ has a unique solution so does e.

Now use that (TA, τ_A) is a free cia on A to obtain a unique H-algebra homomorphism $\overline{\lambda}_A : TA \longrightarrow SA$ extending η^S_A, i. e., such that $\overline{\lambda}_A \cdot \eta_A = \eta^S_A$. One readily proves that $\overline{\lambda}$ is natural in A, that it is a monad morphism from $\mathbb{T}$ to $\mathbb{S}$, that it is uniquely determined, and that it is an ideal monad morphism. In fact, see Theorem 4.4 of [M_1] for the details. □

Theorem 6.9 shows that any iteratable endofunctor admits a free completely iterative monad. The converse is the main result of [M_2], see also [M_1] for an improved proof. This establishes that a free completely iterative monad on an endofunctor is precisely the monad of free cias. Notice that this result does not need any side conditions on $\mathcal{A}$ as is the case in Theorem 2.42 relating free monads and free algebras.

Theorem 6.10. ([M_1], Theorem 6.1)
Every endofunctor generating a free completely iterative monad is iteratable. More precisely, if H has a free completely iterative monad $\mathbb{T} = (T, \eta, \mu, T', t, \mu')$, then H is iteratable, and moreover, for any object A, TA is the final coalgebra for $H(\_) + A$.

Sketch of Proof. Given a free completely iterative monad T on H with a canonical transformation $\kappa : H \longrightarrow T$. Then we obtain a completely iterative monad $HT + Id$, see [M_2], Lemmas 3.3 and 3.5, with an ideal natural transformation $\mathsf{inl} \cdot H\eta : H \longrightarrow HT + Id$. Thus, we obtain a unique ideal monad morphism $\alpha : T \longrightarrow HT + Id$ such that $\alpha \cdot \kappa = \mathsf{inl} \cdot H\eta$. One readily shows that the natural transformation

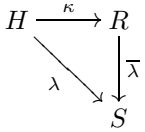
$$\beta \equiv HT + Id \xrightarrow{\kappa + Id} TT + Id \xrightarrow{[\mu, \eta]} T$$

is an inverse of α. Then it is not difficult to show that TA is a final coalgebra for $H(\_) + A$. In fact, every coalgebra $c : C \longrightarrow HC + A$ yields a guarded equation morphism

$$e \equiv C \xrightarrow{\quad c \quad} HC + A \xrightarrow{[\kappa_C, \eta_A]} TC + TA \xrightarrow{\quad \mathsf{can} \quad} T(C + A)\,.$$

The solutions of e are in one to one correspondence with coalgebra homomorphisms from (C, c) to (TA, α_A). Since there exists a unique solution $e^\dagger$ it follows that (TA, α_A) is the desired final coalgebra. □

Theorem 6.11. ([AMV$_2$], Theorem 5.13)
Let H be a finitary endofunctor of an lfp category $\mathcal{A}$. Then the rational monad $\mathbb{R}$ is the free iterative monad on H. That is, the canonical transformation $\kappa = \varrho \cdot H\eta : H \longrightarrow \mathbb{R}$, see Example 6.5, has the following universal property: for every iterative monad $\mathbb{S}$ and every ideal natural transformation $\lambda : H \longrightarrow S$ there exists a unique ideal monad morphism $\overline{\lambda} : \mathbb{R} \longrightarrow \mathbb{S}$ such that the triangle



commutes. Furthermore, this $\overline{\lambda}$ is an ideal monad morphism.

Remark 6.12. The proof of Theorem 6.11 is analogous to the proof of Theorem 6.9. Furthermore, Theorem 6.11 has the following equivalent, and more categorical, formulation. Let $\mathcal{A}$ be an lfp category and recall that $\mathsf{Fin}[\mathcal{A}, \mathcal{A}]$ is the category of all finitary endofunctors of $\mathcal{A}$ and natural transformations between them. For the category

$$\mathsf{FIM}(\mathcal{A})$$

of all finitary iterative monads (i.e., iterative monads $(S, \eta, \mu, S', \sigma, \mu')$ with S and S' finitary) and ideal monad morphisms we have a forgetful functor

$$U : \mathsf{FIM}(\mathcal{A}) \longrightarrow \mathsf{Fin}[\mathcal{A}, \mathcal{A}], \qquad \mathbb{S} \longmapsto S' .$$

Theorem 6.11 states that U has a left adjoint, viz, the functor $H \longmapsto \mathbb{R}$.

A similar formulation of Theorem 6.9 is possible when $\mathcal{A}$ is a locally presentable category, see Remark 2.13, and only accessible functors are considered, see [M$_1$], Remark 4.4, for the precise formulation.

Remark 6.13. Observe that in case of a polynomial endofunctor H_Σ of Set the results of Theorems 6.9 and 6.11 specialize to second-order substitution of all (rational) Σ-trees. In fact, recall Example 2.44, and notice that for signatures Σ and Γ a natural transformation $\lambda : H_\Sigma \longrightarrow T_\Gamma$ essentially gives an "implementation" to each operation symbol of Σ as a Γ-tree. When infinite trees are involved there is usually the restriction to so-called *non-erasing* substitutions; i. e., all Σ-symbols are assigned to trees which are not just single node trees labelled by a variable. That means that λ is an ideal natural transformation. The action of the induced monad morphism $\overline{\lambda} : T_\Sigma \longrightarrow T_\Gamma$ is to perform for every set X the second-order substitution of trees of $T_\Sigma X$ according to λ_X. The result of Theorem 6.11 now implies that if for all sets X, the image of λ_X consists of rational trees of $R_\Gamma X$ then the above monad morphism has a domain-codomain restriction to $\overline{\lambda} : R_\Sigma \longrightarrow R_\Gamma$, i. e., the result of second-order substitution according to λ applied to rational Σ-tree is a rational Γ-tree. Shortly, rational trees are closed under second-order substitution.

7 Recursive Program Schemes

Describe, using diagrams where appropriate,
the exact circumstances leading to your death.
Grant Naylor, Red Dwarf: Infinity Welcomes Careful Drivers.

In this section we present the results of our paper [MM], which is joint work with Larry Moss. Here we shall be interested in the semantics of recursive program schemes such as the one given in (1.7). Their classical theory is compactly presented in [G]. Recall that we distinguish between uninterpreted and interpreted semantics of recursive program schemes. For example, let Σ be a signature of givens consisting of a binary symbol F and a unary one G. Then (1.7) is an uninterpreted recursive program scheme and its semantics is given by the infinite trees in (1.8), which are the result of unfolding the given scheme. An interpreted scheme comes with an algebra having operations for all the givens. A basic example is the recursive program scheme (1.9) defining the factorial function. The standard way to obtain this interpreted solution is to turn the natural numbers into a continuous algebra, and to compute the least fixed point of the continuous function induced by (1.9).

In this section we shall present a generalization of the classical theory based on the results of the previous sections. The point in a nutshell is that knowing that Σ-trees (over a set X) are the final coalgebra for a polynomial functor on Set allows us to provide an uninterpreted semantics to recursive program schemes, i. e., we show how to study and solve (suitably generalized) recursive program schemes in final coalgebras. A part of the proof of our Solution Theorem 7.9 below is inspired by the work of Neil Ghani, Christoph Lüth and Federico De Marchi, see [GLM$_2$]. They have obtained a general solution theorem with the aim of providing a categorical treatment of uninterpreted recursive program scheme solutions. However, the connection we provide to (generalized) second-order substitution in Theorem 6.9 is new in our work.

Furthermore, we present an interpreted semantics of recursive program schemes. We show how to give an interpreted solution to recursive program schemes in arbitrary complete Elgot algebras as introduced in Section 4. Recall that in the classical theory a fundamental result states that uninterpreted and interpreted solutions are consistent. We explained this in the introduction, see (1.12). In our categorical treatment of recursive program schemes we can formulate this precisely, and we shall see that it follows immediately from the proofs of our solution theorems for (un)interpreted program schemes.

Finally, we present several applications of our theory. Our method for obtaining interpreted solutions easily specializes to the usual denotational semantics using complete partial orders. We also show how to solve recursive program schemes in complete metric spaces, see Examples 1.13 and 1.14 in the introduction. Furthermore, we obtain application which go beyond the possibilities of the classical theory. For example, it is possible to recursively define operations satisfying certain equations like commutativity with a recursive program scheme directly in out setting.

7.1 Iteratable Endofunctors

Assumption 7.1. For the rest of Section 7 we assume that $\mathcal{A}$ is a category with finite coproducts (having monomorphic injections). In addition all endofunctors on $\mathcal{A}$ we consider are assumed to be iteratable, see Definition 3.12.

There are fairly mild conditions. In fact, monomorphic coproduct injections could even be avoided, see our discussion after Assumption 6.1. However, we admit that iteratability is not a very nice notion with respect to closure properties of functors—for example, iteratable functors need not be closed under coproducts or composition, see [AAMV], Example 2.12. In the concrete categories we consider here there are stronger yet much nicer conditions that ensure iteratability:

Examples 7.2.

(i) Recall from Remark 2.13 that in the category Set an endofunctor is called accessible if it preserves λ-filtered colimits for some regular cardinal λ. Every accessible endofunctor is iteratable, see [B$_3$]. In particular, functors derived from signatures on Set are iteratable. We discussed the final coalgebra $T_\Sigma X$ of $H_\Sigma(\_) + X$ in Example 2.29(vi).

(ii) Recall from Example 4.11(i) the category CPO of ω-complete partial orders and continuous functions. Unfortunately, not all locally continuous functors of CPO need be iteratable. For a counterexample consider the endofunctor assigning to a cpo X the power set of the set of order components of X. This

is a locally continuous endofunctor but it does not have a final coalgebra. However, every accessible endofunctor H of CPO has a final coalgebra, see [B_3], and moreover, H is iteratable.

(iii) Recall the category CMS of complete metric spaces from Example 3.6(vi). Every contracting endofunctor on CMS is iteratable, see [ARe].

Notation 7.3. As we shall frequently have to deal with final coalgebras coming from two different functors we denote from now on for an endofunctor H of $\mathcal{A}$ by

$$\mathcal{T}(H)X$$

the final coalgebra for $H(\_) + X$ (or, equivalently, a free cia on X). Whenever confusion is unlikely we will drop the parenthetical (H) and simply write TX as before. Recall from Corollary 3.13 that $\mathcal{T}(H)$ carries the structure of a monad whose unit and multiplication we denote by

$$\eta^H : Id \longrightarrow T \qquad \text{and} \qquad \mu^H : TT \longrightarrow T\,,$$

respectively. Recall furthermore, that the structures of the free cias TX yield a natural transformation

$$\tau^H : HT \longrightarrow T\,,$$

and from Example 6.5 we have the canonical natural transformation

$$\kappa^H \equiv H \xrightarrow{H\eta^H} HT \xrightarrow{\tau^H} T\,.$$

We shall frequently drop the superscripts if confusion is unlikely.

Examples 7.4. We mention additional examples of iteratable endofunctors of interest.

(i) Recall from Example 3.11(iii) that every finitary endofunctor H of Set comes as a quotient $\varepsilon : H_\Sigma \longrightarrow H$ of some polynomial functor H_Σ and that its free cias TX are given as a quotient of Σ-trees over X modulo finite and infinite application of the basic equations describing ε.

(ii) The free cias of the finite power set functor $\mathcal{P}_{\mathrm{fin}}$ are the algebras TY of strongly extensional finitely branching trees over Y, see Example 3.11(ii). There is also a different but very elegant description of TY using non-well founded sets, see [BM]. In fact, assuming the Anti-Foundation Axiom, a final $\mathcal{P}_{\mathrm{fin}}$-coalgebra is carried by the set HF_1 of hereditarily finite sets. Similarly, TY is the set $HF_1(Y)$ of hereditarily finite sets generated from the set Y.

(iii) In our applications, the key point is that certain Set functors lift to (iteratable) endofunctors of CPO. And we need to know that those liftings are locally continuous. In fact, let H be an iteratable Set functor with a locally continuous lifting H' on CPO, see Example 4.11(iii). Then H' is iteratable, and moreover, the final coalgebra functor $\mathcal{T}(H')$ is a lifting of $\mathcal{T}(H)$:

$$\begin{array}{ccc} \mathsf{CPO} & \xrightarrow{\mathcal{T}(H')} & \mathsf{CPO} \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathsf{Set} & \xrightarrow[\mathcal{T}(H)]{} & \mathsf{Set} \end{array} \tag{7.39}$$

To see this first recall that for every set X the final coalgebra $\mathcal{T}(H)X$ is obtained from the final coalgebra chain T_i of $H(\_) + X$, see Theorem 2.32. In fact, $\mathcal{T}(H)X$ is the coalgebra $(T_j, t_{j+1,j}^{-1})$ for the smallest ordinal number j for which $t_{j+1,j}$ is bijective. Since the forgetful functor U preserves limits, it follows that for a cpo X the final coalgebra chain of $H'(\_) + X$ has the T_i as underlying sets. However, in CPO the continuous map $t_{j+1,j}$ might not be invertible. But since the chain of underlying sets converges at index j we know that for all ordinal numbers k the connecting maps $t_{j+k,j} : T_{j+k} \longrightarrow T_j$ are monomorphisms of CPO. Moreover, all cpos T_{j+k} have (up to isomorphism) the same underlying set T_j and therefore the partial orders on the T_{j+k}, $k \geq 0$, form a decreasing chain of subsets of $T_j \times T_j$. This implies that the final coalgebra chain has to converge at some index $j + k$, $k \leq \mathsf{card}(T_j \times T_j)$. By standard arguments it follows that the cpo T_{j+k} is the final coalgebra of $H'(\_) + X$. Thus, we

may choose $\mathcal{T}(H')X = T_j$ equipped with the cpo structure given by T_{j+k}, whence the square (7.39) commutes as desired.

For example, every polynomial functor H_Σ has a locally continuous and iteratable lifting H'. The lift is the functor

$$H'X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X^2 + \cdots$$

on CPO. Here each Σ_n is a discretely ordered set, $+$ is the coproduct of CPO (a lift of the coproduct on Set) and $\times$ the usual product. It should be noted that even if X has a least element $\bot$, $H'X$ almost never has one. Finally, $\mathcal{T}(H')X$ is the Σ-tree algebra $T_\Sigma X$ with the order induced by the order of the cpo X—we describe this order more detailed later in Example 7.24(i).

(iv) For a set endofunctor H with a contracting lifting H' on CMS, see Example 3.6(vii), we have that H is iteratable and $U \cdot \mathcal{T}(H') = \mathcal{T}(H) \cdot U$, for the forgetful functor $U : \mathsf{CMS} \longrightarrow \mathsf{Set}$. In fact, this follows from the results of [ARe] since U preserves limits. Recall that every polynomial functor H_Σ of Set has a contracting lifting to CMS. The final coalgebra $\mathcal{T}(H')X$ is the Σ-tree algebra $T_\Sigma X$ equipped with a suitable complete metric. We will discuss this metric later in Remark 7.26.

7.2 Uninterpreted Recursive Program Schemes

The classical treatment of recursive program schemes fits into our work in the following way: Suppose we have a signature Σ of *given* operation symbols. Let Φ be a (finite) signature of new operation symbols. Classically a *recursive program scheme* (or shortly, *RPS*) gives for each operation symbol $f \in \Phi_n$ a term t^f over $\Sigma + \Phi$ in n variables. We thus have a system of formal equations

$$f(x_1, \ldots, x_n) \approx t^f(x_1, \ldots, x_n), \qquad f \in \Phi_n, \qquad n \in \mathbb{N}\,. \tag{7.40}$$

Now observe that the names of the variables in (7.40) do not matter. More precisely, regarding Φ as a functor from $\mathbb{N}$ to Set as in Example 2.44, any RPS as in (7.40) gives rise to a natural transformation

$$\Phi \longrightarrow T_{\Sigma+\Phi} \cdot J\,, \tag{7.41}$$

where recall that $J : \mathbb{N} \longrightarrow \mathsf{Set}$ is the inclusion functor mapping a number n to the set $\{\,0, \ldots, n-1\,\}$. The formulation in (7.41) insures that in each equation of an RPS such as (7.40), if the symbol on the left side is n-ary, then the variables that can appear on the right are the n elements of $\{\,0, \ldots, n-1\,\}$. Notice as well that our formulation extends the classical notion of RPS in the sense that by taking $T_{\Sigma+\Phi}$ we allow infinite trees on the right-hand sides. Furthermore, we will generalize this notion of RPS.

The natural transformation in (7.41) corresponds to a unique natural transformation

$$H_\Phi \longrightarrow T_{\Sigma+\Phi}\,. \tag{7.42}$$

as explained in Example 2.44. The point is that the formulation in (7.42) is more useful to us than the one in (7.41) because (7.42) involves a natural transformations between endofunctors on one and the same category.

Now notice that $T_{\Sigma+\Phi} = \mathcal{T}(H_\Sigma + H_\Phi)$, where H_Σ and H_Φ denote the polynomial functors for Σ and Φ, respectively. With this in mind, we can rewrite (7.42), and we see that recursive program schemes correspond to natural transformations of the following form:

$$H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)\,.$$

Example 7.5. Let Σ contain a unary operation symbol G and a binary one F. The signature Φ of recursively defined operations contains two unary symbols φ and ψ. Consider the recursive program scheme (1.7). The polynomial functor expressing the givens is $H_\Sigma = X + (X \times X)$ and the recursively defined operations Φ are expressed by $H_\Phi X = X + X$. Thus, the scheme (1.7) gives a natural transformation $H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$.

Similarly, the RPS (1.9) defining the factorial function with the signature Σ of givens and the signature Φ containing the unary symbol f gives rise to a natural transformation $H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$.

In the classical treatment of recursive program schemes one gives an uninterpreted semantics to systems like (1.7) which are in Greibach normal form; i. e., every term on the right-hand side of the system has as its head symbol a symbol from the signature Σ of givens. The semantics assigns to each of the new operation symbols a tree over Σ. These trees are obtained as the result of unfolding the recursive specification of the

RPS. But much has been omitted so far. We have given no general account of any solution method, or even, why the trees in (1.8) solve the scheme (1.7).

We will now abstract away from signatures and sets and study the uninterpreted and the interpreted semantics of recursive program schemes considered as natural transformations of the form $V \longrightarrow \mathcal{T}(H+V)$ where H, V, and $H+V$ are iteratable endofunctors on the category $\mathcal{A}$. To say what a solution of a recursive program scheme is we use the universal property of the monads $\mathcal{T}(H)$ as presented in Theorem 6.9. It gives an abstract version of second-order substitution. Here are our central definitions, generalizing recursive program schemes from signatures to completely iterative monads.

Definition 7.6. Let V and H be endofunctors on $\mathcal{A}$. A *recursive program scheme* (or *RPS*, for short) is a natural transformation

$$e : V \longrightarrow \mathcal{T}(H+V) \ .$$

We sometimes call V the *variables*, and H the *givens*.

The RPS e is called *guarded* if there exists a natural transformation $f : V \longrightarrow H\mathcal{T}(H+V)$ such that the following diagram commutes:

$$\begin{array}{ccc} V & \xrightarrow{\ e\ } & \mathcal{T}(H+V) \\ & & \uparrow{\scriptstyle \tau^{H+V}} \\ & & (H+V)\mathcal{T}(H+V) \\ {\scriptstyle f}\downarrow & & \uparrow{\scriptstyle \mathsf{inl}*\mathcal{T}(H+V)} \\ & \searrow & H\mathcal{T}(H+V) \end{array} \tag{7.43}$$

A *solution* of e is an ideal natural transformation $e^\dagger : V \longrightarrow \mathcal{T}(H)$ such that the following triangle commutes:

$$\begin{array}{ccc} V & \xrightarrow{\ e^\dagger\ } & \mathcal{T}(H) \\ {\scriptstyle e}\downarrow & \nearrow_{\overline{[\kappa^H, e^\dagger]}} & \\ \mathcal{T}(H+V) & & \end{array} \tag{7.44}$$

Remark 7.7. Recall that $\overline{[\kappa^H, e^\dagger]}$ is the unique ideal monad morphism extending $\sigma = [\kappa^H, e^\dagger] : H+V \longrightarrow \mathcal{T}(H)$, see Theorem 6.9. Observe that therefore it is important to require that $e^\dagger$ be an ideal natural transformation since otherwise $\overline{\sigma}$ is not defined.

Remark 7.8.

(i) From the discussion at the beginning of this subsection it follows that our definition is a generalization of the classical notion of RPS (to the category theoretic setting), and it extends the classical work as well by allowing infinite trees on the right-hand sides of equations.

Our notion of guardedness captures precisely the requirement that all right-hand sides of (7.40) have their root labelled by a symbol from the givens Σ. In the classical treatment of RPS this is precisely what is called Greibach normal form of an RPS, see [C].

Notice that the two recursive program schemes of Example 7.5 are in Greibach normal form, whence they give rise to guarded RPS in the sense of Definition 7.6.

(ii) Suppose that $H = H_\Sigma$ and $V = H_\Phi$ are polynomial endofunctors of Set, and consider the recursive program scheme $e : H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$ as a set of formal equations as in (7.40). Then for every set X of syntactic variables the X-component $e^\dagger_X : H_\Phi X \longrightarrow T_\Sigma X$ of a solution assigns to a flat tree $(f, x_1, \ldots, x_n) = (f, \vec{x})$ from $H_\Phi X$ a Σ-tree over X. The commutativity of the triangle (7.44) gives the following property of solutions: apply to the right-hand side $t^f(\vec{x})$ of $f(\vec{x})$ in the given RPS the second-order substitution that replaces each operation symbol of Φ by its solution, and each operation symbol of Σ by itself—this is the action of $\overline{[\kappa^H, e^\dagger]}_X$. The resulting tree in $T_\Sigma X$ is the same tree as $e^\dagger_X(f, \vec{x})$.

Theorem 7.9. ([MM], Theorem 6.15)
Every guarded recursive program scheme has a unique solution.

Sketch of Proof. Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be any iteratable functor. Then $T = \mathcal{T}(H)$ is a final coalgebra for the functor $H \cdot \_ + Id$ on the endofunctor category $[\mathcal{A}, \mathcal{A}]$. This result extends to the comma-category $H/\mathsf{M}(\mathcal{A})$ whose objects are pairs $(S, \sigma : H \longrightarrow S)$ where S is a monad on $\mathcal{A}$ and σ is a natural transformation, and whose morphisms $h : (S_1, \sigma_1) \longrightarrow (S_2, \sigma_2)$ are monad morphisms $h : S_1 \longrightarrow S_2$ with $h \cdot \sigma_1 = \sigma_2$. In fact, we obtain a functor $\mathcal{H}$ on $H/\mathsf{M}(\mathcal{A})$ with $\mathcal{H}(S, \sigma) = (HS + Id, \mathsf{inl} \cdot H\eta)$, where $\eta : Id \longrightarrow S$ is the unit of the monad S. This functor restricts to the subcategory of $H/\mathsf{M}(\mathcal{A})$ formed by all pairs (S, σ), where S is a completely iterative monad and σ an ideal natural transformation, and by all ideal monad morphisms, see 6.4. Furthermore, T together with $\kappa^H : H \longrightarrow T$, see Notation 7.3, is the final $\mathcal{H}$-coalgebra.

Now suppose we are given a guarded RPS $e : V \longrightarrow \mathcal{T}(H + V)$. Then we have a natural transformation $\sigma = \mathsf{inl} \cdot [H\eta^{H+V}, f] : H + V \longrightarrow H\mathcal{T}(H+V) + Id$ which is ideal in the sense that it factors through $H\mathcal{T}(H+V)$. Notice that $H\mathcal{T}(H + V) + Id$ is obtained by applying the functor $\mathcal{H}$ to $(\mathcal{T}(H + V), \kappa^{H+V} \cdot \mathsf{inl})$. Thus it is a monad; in fact, it is a completely iterative monad. Use the universal property of the free completely iterative monad $\mathcal{T}(H + V)$, see Theorem 6.9, to obtain a monad morphism $\overline{e} : \mathcal{T}(H + V) \longrightarrow H\mathcal{T}(H + V) + Id$. It is easy to see that this gives rise to a $\mathcal{H}$-coalgebra, and so there exists a unique coalgebra homomorphism h from this coalgebra to the final one carried by $(\mathcal{T}(H), \kappa^H)$, which gives a monad morphism. Now define

$$e^\dagger \equiv \; V \xrightarrow{\mathsf{inr}} H + V \xrightarrow{\kappa^{H+V}} \mathcal{T}(H + V) \xrightarrow{h} \mathcal{T}(H)\,.$$

A non-trivial proof shows that this is indeed the desired unique solution of e, see [MM] for details. □

Remark 7.10. The first part of the above proof showing the finality of $\mathcal{T}(H)$ and defining the monad morphism h uses similar ideas than the proof of the main result of [GLM$_2$]. However, the second part in which it is proved that $e^\dagger$ is a solution of e and that this solution is unique is new in our work. It connects solutions to the (generalized) second-order substitution as presented in Theorem 6.9.

Remark 7.11. In the leading example of a classical RPS for given signatures, the formation of the morphism $\overline{e}$ corresponds to the formation of a flat system of equations, where for every (finite and infinite) tree there is a recursion variable. More precisely, suppose we have signatures Σ and Φ, and an RPS as in (7.40) which is in Greibach normal form. The component of $\overline{e} : \mathcal{T}(H_\Sigma + H_\Phi) \longrightarrow H\mathcal{T}(H_\Sigma + H_\Phi)$ at some set X of syntactic variables is a flat equation morphism $\overline{e}_X : T_{\Sigma+\Phi}X \longrightarrow HT_{\Sigma+\Phi}X + X$. Therefore it can be seen as a set of formal equations, which we will now describe. For every tree $t \in T_{\Sigma+\Phi}X$ we have a variable $\underline{t}$. If t is a single node tree labelled by the syntactic variable $x \in X$ then we have the formal equation

$$\underline{t} \approx x\,,$$

and otherwise we have, for some $n \in \mathbb{N}$ and some $\sigma \in \Sigma_n$,

$$\underline{t} \approx \sigma(\underline{t_1}, \ldots, \underline{t_n})\,,$$

where the tree $s = \sigma(t_1, \ldots, t_n)$ is the result of the following second-order substitution applied to t: every symbol of Φ is substituted by its right-hand side in the given RPS, and every symbol of Σ by itself. Since the given RPS is guarded the head symbol of s is a symbol of Σ for all trees t. Observe that forming the right-hand sides of this system corresponds to the application of one step of Kleene's computation rule, see [G].

Now the component at X of the induced natural transformation $h : \mathcal{T}(H_\Sigma + H_\Phi) \longrightarrow \mathcal{T}(H_\Sigma)$ assigns to every $\underline{t} \in T_{\Sigma+\Phi}X$ of the flat system given by $\overline{e}_X$ its solution in the cia $T_\Sigma X$, i. e., the Σ-tree obtained by unfolding the recursive definition of $\underline{t}$ in the flat system. Thus, to every element $(f, \vec{x})$ of $H_\Phi X$ the component of the uninterpreted solution $e^\dagger$ at X assigns the tree unfolding of $\underline{(f, \vec{x})}$ according to the system given by $\overline{e}_X$.

Examples 7.12.

(i) For the guarded RPS of (1.7) the flat system obtained from $\overline{e}_X$ for $X = \{\, x \,\}$ includes the equations of the system

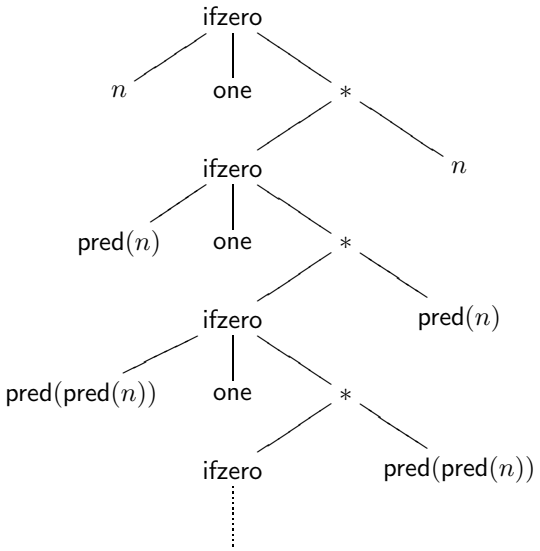
$$\begin{array}{rcl@{\qquad}rcl}
\underline{x} & \approx & x & \underline{\varphi(Gx)} & \approx & F(\underline{Gx}, \underline{\varphi(GGx)}) \\
\underline{\varphi(x)} & \approx & F(\underline{x}, \underline{\varphi(Gx)}) & \underline{F(x, \varphi(Gx))} & \approx & F(\underline{x}, \underline{F(Gx, \varphi(GGx))}) \\
\underline{\psi(x)} & \approx & F(\underline{\varphi(Gx)}, \underline{GGx}) & \underline{GGx} & \approx & G(\underline{Gx}) \\
\underline{\varphi(\psi(x))} & \approx & F(\underline{F(\varphi(Gx, GGx))}, \underline{\varphi(G(F(\varphi(Gx), GGx)))}) & & & \\
 & \vdots & & & &
\end{array}$$

It is clear that the solution of this system in the cia $T_\Sigma X$ will assign to $\underline{\varphi(x)}$ and $\underline{\psi(x)}$ the trees of (1.8).

(ii) The recursive program scheme of (1.9) yields for $X = \{ n \}$ the extension $\overline{e}$ which at X gives a system including the following formal equations:

$$\begin{aligned} \underline{f(n)} &\approx \mathsf{ifzero}(\underline{n}, \underline{\mathsf{one}}, \underline{f(\mathsf{pred}(n)) * n}) \\ \underline{n} &\approx n \\ \underline{\mathsf{one}} &\approx \mathsf{one} \\ \underline{f(\mathsf{pred}(n)) * n} &\approx \underline{\mathsf{ifzero}(\mathsf{pred}(n), \mathsf{one}, f(\mathsf{pred}(\mathsf{pred}(n)))) * \underline{n}} \\ \underline{f(\mathsf{pred}(n))} &\approx \mathsf{ifzero}(\underline{\mathsf{pred}(n)}, \underline{\mathsf{one}}, \underline{f(\mathsf{pred}(\mathsf{pred}(n)) * \mathsf{pred}(n))}) \\ &\vdots \end{aligned} \tag{7.45}$$

The map h_X assigns to $\underline{f(n)}$ the following infinite tree



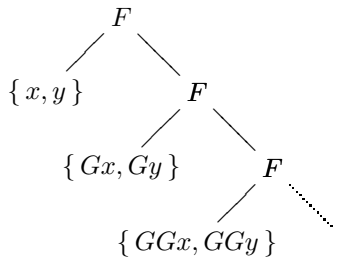
Notice that the nodes labelled by a term correspond to appropriately labelled finite subtrees.

Example 7.13. Let us now present an example of RPSs which are *not* captured in the classical setting. Sometimes one might wish to recursively define new operations from old ones where the new operations should satisfy certain extra properties automatically. We demonstrate this with an RPS defining recursively a new operation which is commutative. Suppose the signature Σ of givens consists of a ternary symbol F and a unary one G. Let us assume that we want to require that F satisfies the equation $F(x, y, z) = F(y, x, z)$ in any interpretation. Then Σ is modelled by the endofunctor $HX = X^3/\!\sim + X$ where $\sim$ is the smallest equivalence on X^3 with $(x, y, z) \sim (y, x, z)$. To be an H-algebra is equivalent to being an algebra A with a unary operation G_A and a ternary one F_A satisfying $F_A(x, y, z) = F_A(y, x, z)$. Suppose that we want to define a commutative binary operation φ by the formal equation

$$\varphi(x, y) \approx F(x, y, \varphi(Gx, Gy)) \,. \tag{7.46}$$

To do it we express φ by the endofunctor V assigning to a set X the set of unordered pairs of X. It is not difficult to see that the formal equation (7.46) gives rise to a guarded RPS $e : V \longrightarrow \mathcal{T}(H+V)$. In fact, to see the naturality use the description of the terminal coalgebra $\mathcal{T}(H + V)Y$ given in Example 3.11(iii). Notice that in the classical setting we are unable to demand that (the solution of) φ is a commutative operation at this stage: this fact would be proved separately once a solution has been obtained. Here we have encoded this additional requirement into our RPS—every solution will be commutative. In fact, the components of

the uninterpreted solution $e^\dagger_X : VX \longrightarrow \mathcal{T}(H)X$ assign to an unordered pair $\{\, x, y \,\}$ in VX the tree



where for every node labelled by F the order of the first two children cannot be distinguished; we indicate this with set-brackets in the picture above.

7.3 Interpreted Recursive Program Schemes

We have seen in the previous section that for every guarded recursive program scheme we can find a unique uninterpreted solution. In practice, however, one is more interested in finding *interpreted* solutions. In the classical treatment of RPS this means that a recursive program scheme defining new operation symbols of a signature Φ from given ones in a signature Σ comes together with some Σ-algebra A. An interpreted solution of the recursive program scheme in question is, then, an operation on A for each operation symbol in Φ such that the formal equations of the RPS become valid equations in A.

Of course, in general an algebra A will not admit interpreted solutions. We shall show in this section that any complete Elgot algebra $(A, a, (\_)^*)$ admits an interpreted solution of every guarded recursive program scheme. Moreover, if A is a cia, interpreted solutions are unique. We also show that uninterpreted solutions and interpreted ones are consistent as explained informally in (1.12). This is a fundamental result for algebraic semantics.

After having presented our main results we turn to applications. In Subsection 7.3.1 we prove that in the category CPO our interpreted program scheme solutions agree with the usual denotational semantics obtained by computing least fixed points. Similarly, we show in Subsection 7.3.2 for the category CMS that our solutions are the same as the ones computed using Banach's Fixed Point theorem. Furthermore, we present new examples of recursive program scheme solutions pertaining to fractal self-similarity.

Notation 7.14. Recall from Theorem 4.17 that for every iteratable functor H the category of complete Elgot algebras for H and their homomorphisms is isomorphic to the category of Eilenberg-Moore algebras for $\mathcal{T}(H)$. Analogously as in Remark 3.15, we denote for any complete Elgot algebra $(A, a, (-)^*)$ by

$$\widetilde{a} : TA \longrightarrow A$$

its associated structure of an Eilenberg-Moore algebra for $T = \mathcal{T}(H)$, and we shall refer to $\widetilde{a}$ as *evaluation morphism* of the complete Elgot algebra $(A, a, (-)^*)$.

Definition 7.15. Let $(A, a, (-)^*)$ be a complete Elgot algebra w.r.t. H and let $e : V \longrightarrow \mathcal{T}(H + V)$ be an RPS. An *interpreted solution* of e in A is a structure of a V-algebra

$$e^\ddagger_A : VA \longrightarrow A\,,$$

such that the $(H + V)$-algebra $[a, e^\ddagger_A] : (H + V)A \longrightarrow A$ is a complete Elgot algebra and such that the triangle

$$\begin{array}{ccc} VA & \xrightarrow{e^\ddagger_A} & A \\ {\scriptstyle e_A}\downarrow & \nearrow_{\widetilde{[a, e^\ddagger_A]}} & \\ \mathcal{T}(H+V)A & & \end{array} \tag{7.47}$$

commutes, where the diagonal arrow denotes the evaluation morphism of the complete Elgot algebra $[a, e^\ddagger_A]$.

Remark 7.16.

(i) In our leading example where $H = H_\Sigma$ and $V = H_\Phi$ are polynomial endofunctors of Set the commutativity of (7.47) states precisely that an interpreted solution provides operations on A which turn the formal equations of the given recursive program scheme into actual identities. More precisely, suppose that e is a recursive program scheme given by formal equations (7.40). The interpreted solution $e^\ddagger_A$ gives for each n-ary operation symbol f of the signature Φ an operation $f_A : A^n \longrightarrow A$, and the commutativity of (7.47) gives the following property of f_A: take for any $\vec{a} \in A^n$ the right-hand side $t^f(\vec{a})$ in the given recursive program scheme, then evaluate $t^f(\vec{a})$ in A using the given operations for Σ and the ones provided by $e^\ddagger_A$ for Φ on A—this is the action of $\widetilde{[a, e^\ddagger_A]}$. The resulting element of A is the same as $f_A(\vec{a})$.

(ii) The requirement that $[a, e^\ddagger_A]$ be the structure morphism of a complete Elgot algebra may seem odd at first. However, we need to assume this in order to be able to use $\widetilde{[a, e^\ddagger_A]}$ in (7.47). Furthermore, the requirement has a clear practical advantage: operations defined recursively by means of an interpreted solution of an RPS may be used in subsequent recursive definitions. For example, for polynomial functors of Set as in (i) above the Elgot algebra with structure map $[a, e^\ddagger_A]$ has operations for all operation symbols of $\Sigma + \Phi$. Thus, it can be used as an interpretation of givens for any further recursive program scheme with signature $\Sigma + \Phi$ of givens.

Theorem 7.17. ([MM], Theorem 7.3)
Let $(A, a, (-)^*)$ be a complete Elgot algebra for H and let $e : V \longrightarrow \mathcal{T}(H + V)$ be a guarded RPS. Then the following hold:

(i) there exists an interpreted solution $e^\ddagger_A$ of e in A,

(ii) if A is a completely iterative algebra, then $e^\ddagger_A$ is the unique interpreted solution of e in A.

Sketch of Proof. Recall the $\mathcal{H}$-coalgebra structure $\overline{e}$ from the proof of Theorem 7.9. The component at A of $\overline{e}$ yields a flat equation morphism

$$\overline{e}_A : \mathcal{T}(H + V)A \longrightarrow H\mathcal{T}(H + V)A + A$$

w.r.t. the given complete Elgot algebra. Denote its solution $(\overline{e}_A)^*$ by β, and define

$$e^\ddagger_A \equiv VA \xrightarrow{\mathsf{inr}} (H + V)A \xrightarrow{\kappa^{H+V}_A} \mathcal{T}(H + V)A \xrightarrow{\beta} A\,. \tag{7.48}$$

A non-trivial proof shows that this morphism is an interpreted solution of e, and that it is the unique interpreted solution if A is a cia. □

Definition 7.18. For any guarded RPS e and any complete Elgot algebra $(A, a, (-)^*)$, let $e^\ddagger_A$ be the interpreted solution obtained from (7.48) above. We call this the *standard interpreted solution* of e in A.

Finally, we state the "Fundamental Theorem of Algebraic Semantics", which establishes that uninterpreted and interpreted solutions are consistent, see (1.12).

Theorem 7.19. ([MM], Theorem 7.7)
Let $(A, a, (\_)^*)$ be a complete Elgot algebra and consider its evaluation morphism $\widetilde{a} : \mathcal{T}(H)A \longrightarrow A$. Let e be any guarded recursive program scheme, let $e^\ddagger_A : VA \longrightarrow A$ be the standard interpreted solution of e in A of Theorem 7.17, and let $e^\dagger : V \longrightarrow \mathcal{T}(H)$ be the (uninterpreted) solution of Theorem 7.9. Then the following triangle commutes:

$$\begin{array}{ccc} VA & \xrightarrow{e^\dagger_A} & \mathcal{T}(H)A \\ & \searrow^{e^\ddagger_A} & \downarrow{\scriptstyle \widetilde{a}} \\ & & A \end{array} \tag{7.49}$$

Furthermore, the standard interpreted solution $e^\ddagger_A$ is uniquely determined by the commutativity of the above triangle.

Remark. Notice that $(e^\dagger)_A$ is the component at A of the natural transformation $e^\dagger$ whereas $e^\ddagger_A$ is not a component of some natural transformation but merely a morphism from VA to A. Also observe that the commuting triangle (7.49) above gives a precise meaning to the informal triangle of (1.12).

Sketch of Proof. Recall the morphisms $h : \mathcal{T}(H+V) \longrightarrow \mathcal{T}(H)$ and $\beta : \mathcal{T}(H+V)A \longrightarrow A$ from the proofs of Theorems 7.9 and 7.17, respectively. It is not difficult to show that the equation $\beta = \widetilde{a} \cdot h_A$ holds. Precompose both sides with $\kappa_A^{H+V} \cdot \mathsf{inr} : VA \longrightarrow \mathcal{T}(H+V)A$ to obtain the desired result.

The uniqueness of $e^\ddagger_A$ follows since neither $e^\dagger_A$ nor $\widetilde{a}$ depend on $e^\ddagger_A$. □

7.3.1 Interpreted Solutions in CPO

We shall show in this subsection that if we have $\mathcal{A} = \mathsf{CPO}$, the category of complete partial orders from Example 4.11(ii), as our base category, then interpreted solutions of guarded RPSs e in a continuous algebra are given as least fixed points of a continuous function on a function space. In this way we recover the usual denotational semantics from our categorical interpreted semantics of recursive program schemes.

Example 7.20. Recall the RPS (1.9), see also Example 7.5. Clearly, the intended interpreted solution is the factorial function on the natural numbers $\mathbb{N}$.

To obtain a suitable complete Elgot algebra in which we can find an interpreted solution of (1.9), we turn the natural numbers into an object of CPO. Let $\mathbb{N}_\perp$ be the flat CPO obtained from the discretely ordered $\mathbb{N}$ by adding a bottom element $\perp$, i. e., $x \leq y$ if and only if $x = \perp$ or $x = y$. We equip $\mathbb{N}_\perp$ with the obvious strict operations $\mathsf{one}_{\mathbb{N}_\perp}$, $\mathsf{pred}_{\mathbb{N}_\perp}$ and $*_{\mathbb{N}_\perp}$, which are all continuous. In addition, we use the continuous operation

$$\mathsf{ifzero}_{\mathbb{N}_\perp}(n,x,y) = \begin{cases} \perp & \text{if } n = \perp \\ x & \text{if } n = 0 \\ y & \text{else} \end{cases}$$

Hence we have a continuous Σ-algebra with $\perp$, and therefore $\mathbb{N}_\perp$ is an Elgot algebra w.r.t. $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$, see Example 4.11(iii).

The interpreted solution $e^\ddagger_{\mathbb{N}_\perp} : H_\Phi \mathbb{N}_\perp \longrightarrow \mathbb{N}_\perp$ will certainly be *some* function or other on $\mathbb{N}_\perp$. But how do we know that this function is the desired factorial function? Usually one would simply regard the RPS (1.9) itself as a continuous function R on $\mathsf{CPO}(\mathbb{N}_\perp, \mathbb{N}_\perp)$ acting as

$$f(\_) \longmapsto \mathsf{ifzero}_{\mathbb{N}_\perp}(\_, 1, f(\mathsf{pred}_{\mathbb{N}}(\_) *_{\mathbb{N}_\perp} \_), \_)$$

That means that we interpret all the operation symbols of Σ in the algebra $\mathbb{N}_\perp$. The usual denotational semantics assigns to the formal equation (1.9) with the interpretation in $\mathbb{N}_\perp$ the least fixed point of R. Clearly this yields the desired factorial function. And it is not difficult to work out that the least fixed point of R coincided with the interpreted solution $e^\ddagger_{\mathbb{N}_\perp}$ obtained from Theorem 7.17. We shall do this shortly in greater generality.

In general, any recursive program scheme can be turned into a continuous function R on the function space $\mathsf{CPO}(VA, A)$. Theorem 7.21 below shows that the least fixed point of R is the same as the interpreted solution obtained from Theorem 7.17.

We assume throughout this subsection that H, V and $H+V$ are locally continuous (and, as always, iteratable) endofunctors of CPO. We also consider a fixed guarded RPS $e : V \longrightarrow \mathcal{T}(H+V)$, and an H-algebra (A, a) with a least element $\perp$. By Example 4.11(ii), we know that this carries the structure of a complete Elgot algebra $(A, a, (-)^*)$, where $(-)^*$ assigns to every flat equation morphism its least solution. As before we will use the notation $\widetilde{a} : \mathcal{T}(H)A \longrightarrow A$ for the evaluation morphism. Furthermore, observe that for every continuous map $f : VA \longrightarrow A$ we have a complete Elgot algebra on A with structure $[a, f] : (H+V)A \longrightarrow A$. Due to Remark 4.18(i), its evaluation morphism satisfies

$$\widetilde{[a,f]} \cdot \kappa_A^{H+V} = [a, f]\,. \tag{7.50}$$

Theorem 7.21. ([MM], Theorem 7.10)
The following function R on $\mathsf{CPO}(VA, A)$

$$f \longmapsto VA \xrightarrow{e_A} \mathcal{T}(H+V)A \xrightarrow{\widetilde{[a,f]}} A \tag{7.51}$$

is continuous. Its least fixed point is the standard interpreted solution $e^\ddagger_A : VA \longrightarrow A$ of Theorem 7.17.

Sketch of Proof. To see the continuity of R is suffices to prove that $\widetilde{(\_)} : \mathsf{CPO}(HA, A) \longrightarrow \mathsf{CPO}(\mathcal{T}(H)A, A)$ is continuous. Let us write T for $\mathcal{T}(H)$. Recall from Remark 4.18 that for every continuous map $a : HA \longrightarrow A$ the evaluation morphism $\widetilde{a}$ is the least solution of the flat equation morphism $\alpha_A : TA \longrightarrow HTA + A$, i. e., $\widetilde{a}$ is the least fixed point of the continuous function $F(a, -) : \mathsf{CPO}(TA, A) \longrightarrow \mathsf{CPO}(TA, A)$ with $F(a, f) = [a, A] \cdot (Hf + A) \cdot \alpha_A$. Observe that F is continuous in the first argument a, and so F is a continuous function on the product $\mathsf{CPO}(HA, A) \times \mathsf{CPO}(TA, A)$. It follows from standard arguments that taking the least fixed point in the second argument yields a continuous map $\mathsf{CPO}(HA, A) \longrightarrow \mathsf{CPO}(TA, A)$. But this is precisely the desired one $\widetilde{(\_)}$.

We prove that $e^\dagger_A$ is the least fixed point of R. Notice that the least fixed point of R is the join t of the increasing chain in $\mathsf{CPO}(VA, A)$ given by $t_0 = \mathsf{const}_\perp$ and $t_{i+1} = \widetilde{[a, t_i]} \cdot e_A$, for $i \in \mathbb{N}$.

Furthermore, recall that the interpreted solution $e^\dagger_A$ is defined by $\beta \cdot \kappa_A^{H+V} \cdot \mathsf{inr}$, where $\beta = (\overline{e}_A)^*$ is the least solution of the flat equation morphism which is obtained from the component at A of the $\mathcal{H}$-coalgebra $\overline{e}$, see Theorem 7.17. By Example 4.11(ii), the solution β of $\overline{e}_A$ is the join of the chain given by $\beta_0 = \mathsf{const}_\perp$ and $\beta_{i+1} = [a, A] \cdot H(\beta_i + A) \cdot \overline{e}_A$, for $i \in \mathbb{N}$.

Observe that $e^\dagger_A$ is a fixed point of R, see (7.47). Thus, we have $t \sqsubseteq e^\dagger_A$. To show the reverse inequality one proves by induction on i the inequalities $\beta_i \sqsubseteq \widetilde{[a, t]}$, for $i \in \mathbb{N}$, see [MM]. This implies that $\beta \sqsubseteq \widetilde{[a, t]}$ and therefore $e^\dagger_A = \beta \cdot \kappa_A^{H+V} \cdot \mathsf{inr} \sqsubseteq \widetilde{[a, t]} \cdot \kappa_A^{H+V} \cdot \mathsf{inr} = t$, by (7.50) above. □

Remark 7.22. The result of Theorem 7.21 implies a concrete formula

$$e^\dagger_A = \bigsqcup_{n<\omega} e^\dagger_n$$

for the interpreted solution of the guarded RPS e in the continuous algebra A. In fact, the least fixed point of R is the join of the ascending chain

$$\perp \sqsubseteq R(\perp) \sqsubseteq R^2(\perp) \sqsubseteq \cdots$$

where $\perp = \mathsf{const}_\perp$ is the least element of $\mathsf{CPO}(VA, A)$. Thus, with $e^\dagger_0 = \mathsf{const}_\perp$ and

$$e^\dagger_{n+1} \equiv VA \xrightarrow{e_A} \mathcal{T}(H + V)A \xrightarrow{\widetilde{[a, e^\dagger_n]}} A$$

we obtain the above formula for $e^\dagger_A$.

Remark 7.23. Suppose that H, V and $H + V$ are iteratable endofunctors of Set, which have locally continuous liftings H' and V' to CPO. Then we have a commutative square

$$\begin{array}{ccc} \mathsf{CPO} & \xrightarrow{\mathcal{T}(H'+V')} & \mathsf{CPO} \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathsf{Set} & \xrightarrow[\mathcal{T}(H+V)]{} & \mathsf{Set} \end{array}$$

by Example 7.4(iii). Assume that the guarded RPS $e : V \longrightarrow \mathcal{T}(H + V)$ has a lifting $e' : V' \longrightarrow \mathcal{T}(H' + V')$; i. e., a natural transformation e' such that $U * e' = e * U$. Now consider any CPO-enrichable H-algebra (A, a) as a complete Elgot algebra, see Example 4.11(iii). Then we can apply Theorem 7.21 to obtain an interpreted solution $e^\dagger_A$ of e in the algebra A as a least fixed point of the above function R of (7.51).

Example 7.24.

(i) Suppose we have signatures Σ and Φ. Then the polynomial functors H_Σ and H_Φ have locally continuous liftings H'_Σ and H'_Φ. Since the lifting of $H_\Sigma + H_\Phi$ is a lifting of $H_{\Sigma+\Phi}$ we know that $\mathcal{T}(H'_\Sigma + H'_\Phi)$ assigns to a cpo X the algebra $T_{\Sigma+\Phi}X$ with the cpo structure induced by X, see Example 7.4(iii). More precisely, to compare a tree t to a tree s replace all leaves labelled by a variable from X by a leaf labelled by some extra symbol $\star$ to obtain relabelled trees t' and s'. Then $t \sqsubset s$ holds in $T_{\Sigma+\Phi}X$ if and only if t' and s' are isomorphic as labelled trees, and for any leaf of t labelled by a variable x the corresponding leaf in s is labelled by a variable y with $x \sqsubseteq y$ in X.

Now consider any system as in (7.40) which is in Greibach normal form, and form the associated guarded RPS $e : H_\Phi \longrightarrow T_{\Sigma+\Phi}$. Then e has a lifting $e' : H'_\Phi \longrightarrow \mathcal{T}(H'_\Sigma + H'_\Phi)$. In fact, for every cpo X the component $e'_X = e_X : H_\Phi X \longrightarrow T_{\Sigma+\Phi}X$ is a continuous map since the order in $H_\Phi X$ is given similarly as for $T_{\Sigma+\Phi}X$ on the level of variables only.

Let (A, a) be a CPO-enrichable H_Σ-algebra; i. e., a continuous Σ-algebra with a least element $\bot$. We wish to consider the continuous function R on $\mathsf{CPO}(H_\Phi A, A)$ which assigns to a continuous algebra structure $\varphi : H_\Phi A \longrightarrow A$ the algebra structure $R(\varphi) = \widetilde{[a, \varphi]} \cdot e'_A$. The structure $R(\varphi)$ gives to each n-ary operation symbol f of Φ the operation $t^f_A : A^n \longrightarrow A$ which is obtained as follows: take the term t^f provided by the right-hand side of f in our given RPS, then interpret all operation symbols of Σ in t^f according to the the given algebraic structure a and all operation symbols of Φ according to φ; the action of t^f_A is evaluation of that interpreted term.

Theorem 7.21 states that an interpreted solution $e^\ddagger_A$ of e in the algebra A can be obtained by taking the least fixed point of R; in other words, the interpreted solution $e^\ddagger_A$ gives the usual denotational semantics.

(ii) Apply the previous example to the RPS of (1.9) considered as a natural transformation, see Example 7.5. Then Theorem 7.21 states that the interpreted solution of the RPS (1.9) in the complete Elgot algebra $\mathbb{N}_\bot$ is obtained as the least fixed point of the function R of Example 7.20. That is, the interpreted solution gives the desired factorial function.

(iii) Recall the guarded RPS e from Example 7.13. Consider again the algebra $\mathbb{N}_\bot$ together with the following two operations:

$$F_{\mathbb{N}_\bot}(x, y, z) = \begin{cases} x & \text{if } x = y \\ z & \text{else} \end{cases} \qquad G_{\mathbb{N}_\bot}(x) = \begin{cases} \lfloor \frac{x}{2} \rfloor & \text{if } x \in \mathbb{N} \\ \bot & x = \bot \end{cases}$$

Since the first operation obviously satisfies $F_{\mathbb{N}_\bot}(x, y, z) = F_{\mathbb{N}_\bot}(y, x, z)$ we have defined an H-algebra. It is not difficult to check that the set functor H has a locally continuous lifting H' on CPO and that $\mathbb{N}_\bot$ is a continuous H'-algebra. In fact, the existence of the lifting H' follows from the fact that the unordered pair functor $V : \mathsf{Set} \longrightarrow \mathsf{Set}$ can be lifted to CPO; the lifting assigns to a cpo $(X, \leq)$ the set of unordered pairs with the following order: $\{ x, y \} \sqsubseteq \{ x', y' \}$ if and only if either $x \leq x'$ and $y \leq y'$, or $x \leq y'$ and $y \leq x$. Thus, we have defined a complete Elgot algebra for $H : \mathsf{Set} \longrightarrow \mathsf{Set}$, see Example 4.11(iii). The interpreted solution $e^\ddagger_{\mathbb{N}_\bot} : V\mathbb{N}_\bot \longrightarrow \mathbb{N}_\bot$ is given by one commutative binary operation $\varphi_{\mathbb{N}_\bot}$ on $\mathbb{N}_\bot$. We leave it to the reader to verify that for natural numbers x and y, $\varphi_{\mathbb{N}_\bot}(x, y)$ is the natural number represented by the greatest common prefix in the binary representation of x and y, e. g., $\varphi_{\mathbb{N}_\bot}(12, 13) = 6$. Notice that we do not have to prove separately that $\varphi_{\mathbb{N}_\bot}$ is commutative. The way we have formed the RPS e in Example 7.13 ensures that the interpreted solution must be given by a commutative operation.

(iv) Least fixed points are RPS solutions. Let A be a poset with joins of all subsets which are at most countable, and let $f : A \longrightarrow A$ be a function preserving joins of ascending chains. Take f and binary joins to obtain an algebra structure on A of the polynomial set functor $H_\Sigma X = X + X \times X$ expressing a binary operation symbol F and a unary one G. Obviously, this functor has a lifting $H' : \mathsf{CPO} \longrightarrow \mathsf{CPO}$ and A is a CPO-enrichable algebra, i. e., A is a complete Elgot algebra. Turn the formal equations (1.7) into a recursive program scheme $e : H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$ as demonstrated in Example 7.5. The RPS e has a lifting $e' : V' \longrightarrow \mathcal{T}(H' + V')$, where V' denotes the lifting of H_Φ. The interpreted solution $e^\ddagger_A : V'A \longrightarrow A$ gives two continuous functions φ_A and ψ_A on A. Clearly, we have $\varphi_A(a) = \bigvee_{n \in \mathbb{N}} f^n(a)$, and in particular $\varphi_A(\bot)$ is the least fixed point of f.

7.3.2 Interpreted Solutions in CMS

Recall the category CMS of complete metric spaces from Example 3.6(vi), and let $H, V : \mathsf{CMS} \longrightarrow \mathsf{CMS}$ be contracting endofunctors. We shall show in this subsection that for every guarded RPS $e : V \longrightarrow \mathcal{T}(H + V)$ we can find a unique interpreted solution in every non-empty H-algebra A. More precisely, assume that we have such a guarded RPS e, and let (A, a) be a non-empty H-algebra. Then A is a cia, and in particular it carries the structure of an Elgot algebra. Notice that for every non-expanding map $f : VA \longrightarrow A$ we obtain an algebra structure $[a, f] : (H + V)A \longrightarrow A$, thus we have the induced evaluation morphism

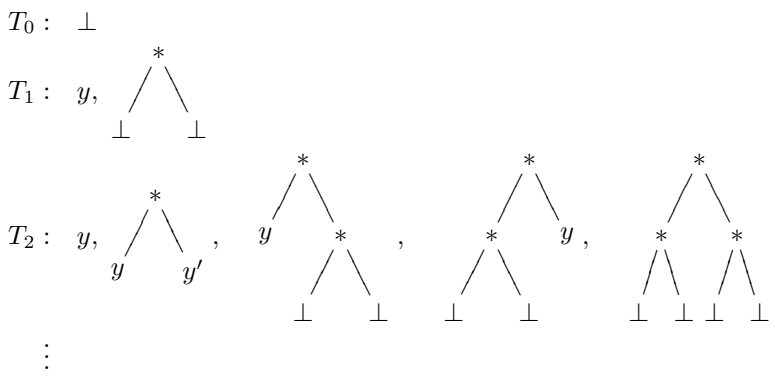
$$\widetilde{[a, f]} : \mathcal{T}(H + V)A \longrightarrow A\,.$$

As in CPO, the RPS e induces a function R on $\mathsf{CMS}(VA, A)$, see (7.51). The standard procedure for obtaining an interpreted solution would be to prove that R is a contracting map, and then invoke Banach's Fixed Point theorem to obtain a unique fixed point of R. Here we simply apply Theorem 7.17. Notice, however, that we cannot completely avoid Banach's Fixed Point theorem: it is used in the proof that final coalgebras exist for contracting functors, see [ARe].

Corollary 7.25. ([MM], Corollary 7.14)
The interpreted solution $e^{\ddagger}_A : VA \longrightarrow A$ of e in A as obtained in Theorem 7.17 is the unique fixed point of the function R on $\mathsf{CMS}(VA, A)$ defined by (7.51).

Proof. In fact, being a fixed point of R is equivalent to being an interpreted solution of e in the cia A, whose unique existence we have by Theorem 7.17. □

Remark 7.26. Let H_Σ be a polynomial functor on Set and denote by H' a contracting lifting to CMS as described in Example 3.6(vii). For a complete metric space Y the final coalgebra $\mathcal{T}(H')Y$ of $H'(\_) + Y$ is the set $T_\Sigma Y$ of all Σ-trees over Y with a suitable complete metric. This metric can be described as follows. Recall from [ARe] that $\mathcal{T}(H')Y$ is obtained as T_ω after ω steps of the final coalgebra chain for $H'(\_) + Y$, see Section 2.3. That means the metric on $T_\Sigma Y$ is the smallest metric such that all projections $t_{\omega,i} : T_\Sigma Y = T_\omega \longrightarrow T_i$ are non-expanding. We illustrate this with an example adapted from [ARe]. Let $H_\Sigma X = X \times X$ be the functor expressing one binary operation symbol $*$. Then we can represent $T_0 = 1$ by a single node tree labelled with $\bot$ and $T_{i+1} = T_i \times T_i + Y$ by trees which are either single node trees labelled in Y, or which are composed by joining two trees from T_i with a root labelled by $*$:



The distance on T_1 is that of Y for single node trees and 1 otherwise. The distance on T_2 is again that of Y between single node trees, and 1 between single node trees and all other trees. Furthermore, the distance between trees of different shapes is $\frac{1}{2}$, and finally, $d_{T_2}(y * y', z * z') = \frac{1}{2}\max\{\, d_Y(y,z), d_Y(y',z') \,\}$ as well as $d_{T_2}(y * t, y' * t) = d_{T_2}(t * y, t * y') = \frac{1}{2} d_Y(y, y')$, where $t = \bot * \bot$, etc. In general, the distance on T_{i+1} is that of Y between single node trees, it is 1 between single node trees and trees of height at least 1, and otherwise we have $d_{T_{i+1}}(s * t, s' * t') = \frac{1}{2}\max\{\, d_{T_i}(s,s'), d_{T_i}(t,t') \,\}$. For the metric on $T_\Sigma Y$, we have

$$d_{T_\Sigma Y}(s_1, s_2) = \sup_{i<\omega} d_{T_i}(t_{\omega,i}(s_1), t_{\omega,i}(s_2)).$$

This is the smallest metric for which the projections are non-expanding. (One may also verify directly that this definition gives a complete metric space structure and that $H'(\_) + Y$ preserves the limit, so that we indeed have a final coalgebra.) Finally notice that the metric of $T_\Sigma Y$ depends on the choice of the lifting H'. For example, if we lift the functor H_Σ as $H'(X,d) = (X^2, \frac{1}{3} d_{\max})$, the factor $\frac{1}{2}$ would have to be replaced by $\frac{1}{3}$ systematically.

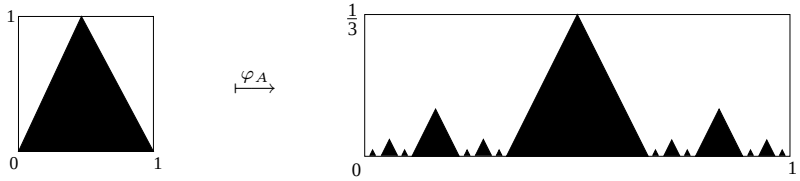
Example 7.27.

(i) Consider the endofunctor $H' : \mathsf{CMS} \longrightarrow \mathsf{CMS}$ obtained by lifting the polynomial set functor $H_\Sigma X = X \times X + X$ expressing a binary operation F and a unary one G as described in Example 3.6(vii). The Euclidean interval $I = [0,1]$ together with the operations $F(x,y) = \frac{x+y}{4}$ and $G(x) = \frac{1}{2}\sin(x)$ is an H'-algebra, whence a cia. Use only the first equation in (1.7) to obtain a guarded RPS $e : Id \longrightarrow \mathcal{T}(H_\Sigma + Id)$ where Id expresses the unary operation symbol φ. Let V' be contracting lifting of Id with a contraction factor of $\varepsilon = \frac{1}{2}$. Then e gives rise to a guarded RPS $e' : V' \longrightarrow \mathcal{T}(H' + V')$ in CMS. The unique interpreted solution of e' in I consists of a function $\varphi_I : I \longrightarrow I$ satisfying $\varphi_I(x) = \frac{1}{4}(x + \varphi_I(\frac{1}{2}\sin x))$, that is, φ_I is the unique function f satisfying (1.13).

(ii) Self similar sets are solutions of interpreted program schemes. Recall from Example 3.6(viii) that for every complete metric space (X, d) we obtain the complete metric space $(C(X), h)$ of all non-empty compact subspaces of X. Furthermore, contractive mappings of X yield structures of cias on $C(X)$. Now consider the functor H' on CMS with $H(X, d) = (X^3, \frac{1}{3}d_{\max})$, where $d_{\max}$ is the maximum metric. It is a lifting of the polynomial functor H_Σ on Set expressing one ternary operation α. Let $A = [0, 1]^2$, be equipped with the usual Euclidean metric. Consider the contracting maps $f(x, y) = (\frac{1}{3}x, \frac{1}{3}y)$, $g(x, y) = (\frac{1}{3}x + \frac{1}{3}, \frac{1}{3}y)$, and $h(x, y) = (\frac{1}{3}x + \frac{2}{3}, \frac{1}{3}y)$ of A. Then it follows that $\alpha_A : C(A)^3 \longrightarrow C(A)$ with $\alpha(D, E, F) = f[D] \cup g[E] \cup h[F]$ is an $\frac{1}{3}$-contracting map, whence a structure of an H'-algebra. The following formal equation

$$\varphi(x) \approx \alpha(\varphi(x), x, \varphi(x))$$

gives rise to a guarded RPS $e : Id \longrightarrow \mathfrak{T}(H_\Sigma + Id)$, where the identity functor expresses the operation φ. If we take the lifting of Id to CMS which is given by $V'(X, d) = (X, \frac{1}{3}d)$, then e gives rise to a natural transformation $e' : V' \longrightarrow \mathfrak{T}(H' + V')$. Its interpreted solution in the algebra $C(A)$ is a $\frac{1}{3}$-contracting map $\varphi_A : C(A) \longrightarrow C(A)$ which maps a non-empty compact subspace U of A to a space of the following form: $\varphi_A(U)$ has three parts, the middle one is a copy of U scaled by $\frac{1}{3}$, and the left-hand and right-hand one look like copies of the whole space $\varphi_A(U)$ scaled by $\frac{1}{3}$. For example we have the assignment



(iii) Coming back to Example 3.6(x) let us consider $(C(I), \alpha_I)$, where $I = [0, 1]$ is the Euclidean interval and α_I is the structure of a cia arising from $fx = \frac{1}{3}x$ and $g(x) = \frac{1}{3}x + \frac{2}{3}$ as described is Example 3.6(viii). The formal equation

$$\varphi(x) \approx \alpha(\varphi(x), x)$$

gives similarly as in (i) above a guarded RPS $e : Id \longrightarrow \mathfrak{T}(H_\Sigma + Id)$, where $H_\Sigma X = X \times X$ now expresses the binary operation α. Again, we have liftings $V'(X, d) = (X, \frac{1}{3}d)$ and $H'(X, d) = (X^2, \frac{1}{3}d_{\max})$ of Id and H_Σ, respectively. So the RPS e lifts to the guarded RPS $e' : V' \longrightarrow \mathfrak{T}(H' + V')$ in CMS. Its unique interpreted solution is given by the $\frac{1}{3}$-contracting map $\varphi_I : C(I) \longrightarrow C(I)$ satisfying $\varphi_I(t) = \alpha_I(\varphi_I(t), t) = f[\varphi_I(t)] \cup g[t]$ for every non-empty closed subset t of the interval I, cf. (1.14).

8 Conclusions and Further Research

> There is a theory which states that if ever anyone discovers exactly what the Universe is for and why it is here, it will instantly disappear and be replaced by something even more bizarre and inexplicable.
> There is another theory which states that this has already happened.
>
> Douglas Adams, *The Restaurant at the end of the Universe.*

In this summary we have presented the material of our papers [AMV_1, AMV_2, AMV_3, AAMV, M_1, M_2, MM]. We have started with the study of iterative and completely iterative algebras for an endofunctor, and we have shown that the free (completely) iterative algebras yield an easy description of free (completely) iterative monads. In this way we have generalized and extended the classical results of Calvin Elgot and his coauthors [E, BE, EBT] following the footsteps of Evelyn Nelson [N]. We have seen that free completely iterative algebras or free completely iterative monads are equivalently given by final coalgebras. We believe that this characterization of final coalgebras as free algebras of some sort is a new and interesting result. In particular, for our subsequent work on semantics of recursive program schemes that connection between coalgebra and algebra is a corner stone—it opens up the possibility to apply coalgebraic methods in algebraic semantics.

In the second part of our work we have introduced and studied (complete) Elgot algebras and we characterized them as the Eilenberg-Moore algebras for free (completely) iterative monads. We then argued that the structure of Elgot algebras captures the most important structural properties of algebras that are suitable for algebraic semantics.

In the last part we have shown how to apply our results to provide semantics of recursive program schemes in a category theoretic way. We have seen that the universal property of free completely iterative monads serves as a generalized second-order substitution and it therefore allows to formulate in a category theoretic way the notion of a solution of a recursive program scheme. We then proved that every guarded recursive program scheme has a unique uninterpreted solution. Next we provided a canonical interpreted semantics of guarded recursive program schemes in complete Elgot algebras.

As the semantics of recursive program schemes is a topic at the heart of theoretical computer science it is an important problem to see whether it can be handled using coalgebraic methods. We are pleased that we can report a success in this matter. Our applications show that we have developed a unified view of solution principles for a large class of recursive definitions including the usual algebraic semantics of recursive program schemes. Of course, our more abstract theory takes somewhat more effort to build. But we feel that the gain in conceptual clarity more than outweighs this small disadvantage.

In addition to the material we have presented here we have obtained a number of results which are closely related but cannot be treated in detail here. We will discuss them briefly now.

Our whole work rests on the assumption that enough final coalgebras for an endofunctor exist. While this is a weak assumption, there are important examples of endofunctors not satisfying this requirement, e. g., the power set functor on Set. In our paper [AMV_4] we have shown that this can be circumvented by working in the category Class of classes in lieu of Set. We proved that every endofunctor of Class has an initial algebra and a final coalgebra so that all endofunctors generate free completely iterative monads. In particular, we gave a description of the final coalgebra for the power set functor. The fact that every endofunctor of Class has a final coalgebra was also independently discovered by Daniela Cancila [Ca]. In [AMV_5] we have extended our work from Set to other categories $\mathcal{K}$. More precisely, we start with any locally small, cocomplete and cowell-powered category $\mathcal{K}$, and we consider its free cocompletion $\mathcal{K}^\infty$ under transfinite colimits, e. g., for $\mathcal{K} = \mathsf{Set}$ we have $\mathcal{K}^\infty = \mathsf{Class}$. We proved that every endofunctor H of $\mathcal{K}$ extends essentially uniquely to an endofunctor H^∞ of $\mathcal{K}^\infty$, and that H^∞ has a final coalgebra. Moreover, if $\mathcal{K}$ is a locally finitely presentable category then H^∞ generates a completely iterative monad in such a way that every guarded equation morphism which lies in the base category $\mathcal{K}$ has its solution in $\mathcal{K}$, too. For example, for the power set functor that means that every guarded equation morphism can be uniquely solved in Set—one does not need to worry about classes at all.

Another line of our current research has been inspired by the work of Tarmo Uustalu [U]. He proposed to study complete iterativity with respect to a so-called base in lieu of an endofunctor. A base is a functor from the category $\mathcal{A}$ to the category of (finitary) monads on $\mathcal{A}$. Bases allow an interesting extension of our theory which captures algebras satisfying certain equations and where the iterativity can be restricted. In fact, we have taken up the task to extend the results on iterative algebras and iterative monads of [AMV_1, AMV_2] presented here to bases with the series of papers [AMV_6, AMV_7, AMV_8, AMV_9, AMV_{10}]. In [AMV_{11}] we present one application of this extended theory; a categorical description of the monad of algebraic trees,

i. e., those trees that arise as uninterpreted solutions of (classical) recursive program schemes, see [C].

Very recently, we have turned our attention to recursive coalgebras, inspired by the work of Venanzio Capretta, Tarmo Uustalu and Varmo Vene [CUV]. A coalgebra is recursive if it admits a unique coalgebra-to-algebra homomorphism into any given algebra. In the paper [ALM] which arose from the second author's diploma thesis, we have proved that for a finitary endofunctor of Set preserving inverse images a coalgebra is recursive if and only if it admits a coalgebra homomorphism into the initial algebra, or if and only if it satisfies the dual concept of complete iterativity. Applications of recursive coalgebras lie, for example, in the realm of the semantics of functional programs using divide-and-conquer strategies such as Quicksort etc. In the subsequent work [AM_2] we will extend the above equivalences beyond finitary functors: the same results hold for all endofunctors of Set preserving inverse images, and moreover, the category Set can be generalized substantially.

Now let us mention a few points which seem worthwhile to be addressed in future research. Our work on recursive program scheme semantics is only in its beginning phase. We suspect that much more can be said about the relation of our work to operational semantics. One should investigate higher-order recursive program schemes using our tools. The paper [MU] addresses variable binding and infinite λ-terms coalgebraically, and this may well be relevant. Back to the classical theory, one of the main goals of the original theory is to serve as a foundation for program equivalence. It is not difficult to prove the soundness of fold/unfold transformations in an algebraic way using our semantics; this was done in [Mo_2] for uninterpreted schemes. One would like more results of this type. The equivalence of interpreted schemes in the natural numbers is undecidable, and so one naturally wants to study the equivalence of interpreted schemes in *classes of interpretations*. The classical theory proposes classes of interpretations, many of which are defined on ordered algebras, see [G]. It would be good to revisit this part of the classical theory to see whether Elgot algebras suggest tractable classes of interpretations.

One of our applications of recursive program schemes showed how to define operations satisfying basic equations, e. g., commutativity. One would like tools to consider more general equations, e. g., associativity. This can however not be achieved by using endofunctors and natural transformations to express recursive program schemes as we have done. One should start with monads expressing givens and variables, e. g., a binary associative operation is expressed by the list monad on Set. But at the moment we are even lacking the most basic tools to study recursive program schemes this way. For example, we need an analogue of a free completely iterative monad on an endofunctor, i. e., we need the completely iterative reflection of a given monad. We have taken first steps in this direction with [M_3].

Another topic for future research is the investigation of the connection of our work to other approaches to semantics, e. g., the traced monoidal categories of [JSV, Ha] or the iteration theories of Stephen Bloom and Zoltán Ésik [BÉ]. For example, the assignment of a free iterative monad to a finitary set endofunctor gives rise to a monad on $\mathsf{Fin}[\mathsf{Set}, \mathsf{Set}]$. We have characterized the Eilenberg-Moore algebras of this monad in [M_4]. We call those Elgot monads, analogously to Elgot algebras. They are monads providing for any guarded equation morphism a canonical solution, i. e., such that certain axioms of the solution operation are fulfilled. These axioms are very similar to the iteration theory axioms. But the precise relationship of Elgot monads and iteration theories needs still to be investigated.

Finally, to conclude this thesis let us sum up the gist of our work in one sentence: We believe to have contributed to the part of the theory of coalgebras which pertains to the semantics of recursion by exploring and using the structure of monads that arises from final coalgebras—in other words, we studied the connection of coalgebras, monads and semantics. Of course, much remains to be done, or as Albert Einstein put it:

The important thing is not to stop questioning.

References

[Ac] Peter Aczel, *Non-Well-Founded Sets*, CSLI Lecture Notes 14, Stanford University, 1988.

[AAV] Peter Aczel, Jiří Adámek and Jiří Velebil, A Coalgebraic View of Infinite Trees and Iteration, *Electron. Notes Theor. Comput. Sci.* 44 (2001), no. 1, 26 pp.

[AAMV] Peter Aczel, Jiří Adámek, Stefan Milius and Jiří Velebil, Infinite Trees and Completely Iterative Theories: A Coalgebraic View, *Theoret. Comput. Sci.* 300 (2003), 1–45.

[A_1] Jiří Adámek, Free Algebras and Automata Realization in the Language of Categories, *Comment. Math. Univ. Carolin.* 15 (1974), 589–602.

[A_2] Jiří Adámek, Introduction the Coalgebra, *Theory Appl. Categ.* 14 (2005), 157–199.

[AK_1] Jiří Adámek and Václav Koubek, Functorial Algebras and Automata, *Kybernetika* 13 (1977), no. 4, 245–260.

[AK_2] Jiří Adámek and Václav Koubek, Are Colimits of Algebras Simple to Construct?, *J. Algebra* 66 (1980), no. 1, 226–250.

[AK_3] Jiří Adámek and Václav Koubek, On the greatest fixed point of a set functor, *Theoret. Comput. Sci.* 150 (1995), 57–75.

[ALM] Jiří Adámek, Dominik Lücke and Stefan Milius, Recursive Coalgebras of Finitary Functors, *submitted*, available via `http://www.iti.cs.tu-bs.de/~milius`.

[AM_1] Jiří Adámek and Stefan Milius, Terminal Coalgebras and Free Iterative Theories, accepted for publication in *Inform. and Comput.*, available via `http://www.iti.cs.tu-bs.de/~milius`.

[AM_2] Jiří Adámek and Stefan Milius, Recursive Coalgebras, *preprint* (2006).

[AMV_1] Jiří Adámek, Stefan Milius and Jiří Velebil, Free Iterative Theories: a coalgebraic view, *Math. Stuctures Comput. Sci.* 13 (2003), no. 2, 259–320.

[AMV_2] Jiří Adámek, Stefan Milius and Jiří Velebil, From Iterative Algebras to Iterative Theories, *submitted*, available via `http://www.iti.cs.tu-bs.de/~milius`, extended abstract appeared in *Electron. Notes Theor. Comput. Sci.* 106 (2004), 3–24.

[AMV_3] Jiří Adámek, Stefan Milius and Jiří Velebil, Elgot Algebras, *preprint* (2005), available via `http://www.iti.cs.tu-bs.de/~milius`, extended abstract to appear in *Electron. Notes Theor. Comput. Sci.*

[AMV_4] Jiří Adámek, Stefan Milius and Jiří Velebil, On Coalgebras based on Classes, *Theoret. Comput. Sci.* 316 (2004), 3–23.

[AMV_5] Jiří Adámek, Stefan Milius and Jiří Velebil, A General Final Coalgebra Theorem, *Math. Structures Comput. Sci.* 15 (2005), no. 3, 409–432.

[AMV_6] Jiří Adámek, Stefan Milius and Jiří Velebil, Algebras with Parameterized Iterativity, *submitted*, available via `http://www.iti.cs.tu-bs.de/~milius`.

[AMV_7] Jiří Adámek, Stefan Milius and Jiří Velebil, Bases for Parameterized Iterativity, *submitted*, available via `http://www.iti.cs.tu-bs.de/~milius`.

[AMV_8] Jiří Adámek, Stefan Milius and Jiří Velebil, Iterative Algebras for a Base, *Electron. Notes Theor. Comput. Sci.* 122 (2005), 147–170.

[AMV_9] Jiří Adámek, Stefan Milius and Jiří Velebil, Base Modules for Parameterized Iterativity, *preprint* (2005).

[AMV_{10}] Jiří Adámek, Stefan Milius and Jiří Velebil, Description of Bases for Parameterized Iterativity, *preprint* (2005).

[AMV_{11}] Jiří Adámek, Stefan Milius and Jiří Velebil, Algebraic Trees Coalgebraically, *manuscript* (2005).

[AP] Jiří Adámek and Hans-Eberhardt Porst, On tree coalgebras and coalgebra presentations, *Theoret. Comput. Sci.* 311 (2004), 257–283.

[ARe] Jiří Adámek and Jan Reitermann, Banach's Fixed-Point Theorem as a Base for Data-Type Equations, *Appl. Categ. Structures* 2 (1994), 77–90.

[AR] Jiří Adámek and Jiří Rosický, *Locally presentable and accessible categories*, Cambridge University Press, 1994.

[AT] Jiří Adámek and Věra Trnková, *Automata and Algebras in Categories*. Kluwer Academic Publishers, 1990.

[ARu] Pierre America and Jan J. M. M. Rutten, Solving Reflexive Domain Equations in a Category of Complete Metric Spaces, *J. Comput. System Sci.* 39 (1989), 343–375.

[AN] André Arnold and Maurice Nivat, The metric space of infinite trees. Algebraic and topological properties, *Fund. Inform.* III, no. 4 (1980), 445–476.

[Ban] Stefan Banach, Sur les opèrations dans les ensembles abstraits et leurs applications aux èquations intègrales, *Fund. Math.* 3 (1922), 133–181.

[Ba] Michael F. Barnsley, *Fractals everywhere*, Academic Press 1988.

[B_1] Michael Barr, Coequalizers and free triples, *Math. Z.* 116, 307–322.

[B_2] Michael Barr, Relational algebras, *Lecture Notes in Math.* 137, Springer Verlag, 39–55.

[B_3] Michael Barr, Terminal coalgebras in well-founded set theory, *Theoret. Comput. Sci.* 114 (1993), 299–315.

[BM] Jon Barwise and Lawrence S. Moss, *Vicious Circles*, CSLI Publications, Stanford, 1996.

[Bl] Stephen L. Bloom, All Solutions of a System of Recursion Equations in Infinite Trees and Other Contraction Theories, *J. Comput. System Sci.* 27 (1983), 225–255.

[BE] Stephen L. Bloom and Calvin C. Elgot, The Existence and Construction of Free Iterative Theories, *J. Comput. System Sci.* 12 (1974), 305–318.

[BÉ] Stephen L. Bloom and Zoltán Ésik, *Iteration Theories: The equational logic of iterative processes,* EATCS Monographs on Theoretical Computer Science, Berlin: Springer-Verlag (1993).

[CUV] Venanzio Capretta, Tarmo Uustalu, Varmo Vene, Recursive Coalgebras from Comonads, *Electron. Notes Theor. Comput. Sci.* 106 (2004), 43–61.

[Ca] Daniela Cancila, Investigations in the Categorical Foundations and Applications of Coalgebras and Hypersets, PhD thesis, University of Udine, 2003.

[C] Bruno Courcelle, Fundamental properties of infinite trees, *Theoret. Comput. Sci.* 25 (1983), no. 2, 95–169.

[Ei] Samuel Eilenberg, The category $\mathcal{C}$. Unpublished manuscript.

[E] Calvin C. Elgot, Monadic Computation and Iterative Algebraic Theories, in: *Logic Colloquium '73* (eds: H. E. Rose and J. C. Shepherdson), North-Holland Publishers, Amsterdam, 1975.

[EBT] Calvin C. Elgot, Stephen L. Bloom and Ralph Tindell, On the Algebraic Structure of Rooted Trees, *J. Comput. System Sci.* 16 (1978), 361–399.

[GU] Peter Gabriel and Friedrich Ulmer, *Lokal präsentierbare Kategorien*, Lecture N. Math. 221, Springer-Verlag, Berlin 1971.

[GLM_1] Neil Ghani, Christoph Lüth and Federico De Marchi, Coalgebraic Monads, *Electron. Notes Theor. Comput. Sci.* 65 (2002), no. 1, 21 pp.

[GLM_2] Neil Ghani, Christoph Lüth and Federico De Marchi, Solving Algebraic Equations using Coalgebra, *Theor. Inform. Appl.* 37 (2003), 301–314.

[GLMP$_1$] Neil Ghani, Christoph Lüth, Federico De Marchi and A. John Power, Algebras, coalgebras, monads and comonads, *Electron. Notes Theor. Comput. Sci.* 44 (2001), no. 1, 18 pp.

[GLMP$_2$] Neil Ghani, Christoph Lüth, Federico De Marchi and A. John Power, Dualising initial algebras, *Math. Structures Comput. Sci.* 13 (2003), no. 2, 349–370.

[GUs] Neil Ghani and Tarmo Uustalu, Coproducts of ideal monads, *Theor. Inform. Appl.* 38 (2004), no. 4, 321–342.

[Gi$_1$] Susanna Ginali, *Iterative Algebraic Theories, Infinite Trees, and Program Schemata*, PhD thesis, University of Chicago, 1976.

[Gi$_2$] Susanna Ginali, Regular trees and the free iterative theory, *J. Comput. System Sci.* 18 (1979), 228–242.

[GTWW] Joseph Goguen, James W. Thatcher, Eric G. Wagner and Jesse B. Wright, Initial algebra semantics and continuous algebras, *J. ACM* 24 (1977), 68–95.

[G] Irène Guessarian, *Algebraic Semantics*. Lecture Notes in Comput. Sci. 99, Springer, 1981.

[Gu] Heinz-Peter Gumm, Elements of the General Theory of Coalgebras, LUATCS'99, Rand Africaans University, Johannesburg, South Africa, 1999.

[Ha] Masahito Hasegawa, Recursion from Cyclic Sharing: Traced Monoidal Categories and Models of Cyclic Lambda Calculi, *Proc. 3rd International Conference on Typed Lambda Calculi and Applications*, Lecture Notes in Comput. Sci. 1210, 196–213 Springer-Verlag, Berlin, 1997.

[JR] Bart Jacobs and Jan J. M. M. Rutten, A Tutorial on (Co)Algebras and (Co)Induction, *Bull. Eur. Assoc. Theor. Comput. Sci. EATCS* 62 (1997), 222–259.

[JSV] André Joyal, Ross Street and Dominic Verity, Traced Monoidal Categories, *Math. Proc. Cambridge Philos. Soc.* 119 (1996), no. 3, 447–468.

[L] Joachim Lambek, A fixpoint theorem for complete categories, *Math. Z.* 103 (1968), 151–161.

[La] F. William Lawvere, *Functorial Semantics of Algebraic Theories*, PhD thesis, Columbia University, 1963.

[Le$_1$] Tom Leinster, General self-similarity: an overview, e-print math.DS/0411343 v1.

[Le$_2$] Tom Leinster, A general theory of self-similarity I, e-print math.DS/041344.

[Le$_3$] Tom Leinster, A general theory of self-similarity II, e-print math.DS/0411345.

[Li$_1$] Fred E. J. Linton, Some aspects of equational categories, *Proceedings of the Conference on Categorical Algebra*, Springer-Verlag, 1966, 84–94.

[Li$_2$] Fred E. J. Linton, Coequalizers in Categories of Algebras, *Lecture Notes in Math.* 80, Springer-Verlag, New York, 1976.

[ML] Saunders MacLane, *Categories for the working mathematician*, 2nd edition, Springer Verlag, 1998.

[Ma] Ernest G. Manes, *Algebraic Theories*, Springer Verlag, 1976.

[DM] Federico De Marchi, *Monads in Coalgebra*, PhD thesis, University of Leicester, 2003.

[MU] Ralph Matthes and Tarmo Uustalu, Substitution in Non-Wellfounded Syntax with Variable Binding, *Electron. Notes Theor. Comput. Sci.* 82 (2003), no. 1, 15 pp.

[M$_1$] Stefan Milius, Completely Iterative Algebras and Completely Iterative Monads, *Inform. and Comput.* 196 (2005), 1–41.

[M$_2$] Stefan Milius, On Iteratable Endofunctors, *Electron. Notes Theor. Comput. Sci.* 69 (2002), 18 pp.

[M$_3$] Stefan Milius, The Iterative Reflection of an Ideal Monad, *manuscript* (2005).

[M_4] Stefan Milius, Elgot monads, *manuscript* (2005).

[MM] Stefan Milius and Lawrence S. Moss, The Category Theoretic Solution of Recursive Program Schemes, full version, *submitted*, available via `http://www.iti.cs.tu-bs.de/~milius`, extended abstract appeared in the Proceedings of the first conference on Algebra and Coalgebra in Computer Science (CALCO'05), *Lecture Notes in Comput. Sci.* 3629 (2005), 293–312.

[MS] Gregory Moore and Nathan Seiberg, Classical and quantum conformal field theory, *Comm. Math. Phys.* 123 (1989), no. 2, 177–254.

[Mo_1] Lawrence S. Moss, Parametric Corecursion, *Theoret. Comput. Sci.*, 260 (2001), no. 1–2, 139–163.

[Mo_2] Lawrence S. Moss, The Coalgebraic Treatment of Second-Order Substitution and Uninterpreted Recursive Program Schemes, *preprint* (2002). 2002.

[N] Evelyn Nelson, Iterative Algebras, *Theoret. Comput. Sci.* 25 (1983), 67–94.

[P] Gordon D. Plotkin, *Domains*, The "Pisa" notes, available from `http://www.dcs.ed.ac.uk/home/gdp/`, 1983.

[R_1] J. J. M. M. Rutten, *Automata and coinduction (an exercise in coalgebra)*, Technical Report SEN-R9803, CWI, Amsterdam, 1998.

[R_2] Jan J. M. M. Rutten, Universal coalgebra, a theory of systems, *Theoret. Comput. Sci.* 249 (2000), no. 1, 3–80.

[Ta] Alfred Tarski, A lattice Theoretical Fixed Point Theorem and its Applications, *Pacific J. Math.* 5 (1955), 285–309.

[T] Jerzy Tiuryn, Unique Fixed Points vs. Least Fixed Points, *Theoret. Comput. Sci.* 12 (1980), 229–254.

[TR] Danielle Turi and Jan J. M. M. Rutten, On the foundations of final coalgebra semantics: non-well-founded sets, partial orders, metric spaces, *Math. Structures Comput. Sci.* 8 (1998), no. 5, 481–540.

[U] Tarmo Uustalu, Generalizing substitution, *Theor. Inform. Appl.* 37 (2003), no. 4, 315–336.

[W_1] James Worrell, *On Coalgebras and Final Semantics*, PhD thesis, Oxford Unversity Computing Laboratory, 2000.

[W_2] James Worrell, On the final sequence of a finitary set functor, *Theoret. Comput. Sci.* 338 (2005), 184–199.

Appendix

In this appendix we include the following of our papers:

[AMV_1]	Free Iterative Theories: a coalgebraic view
[AMV_2]	From Iterative Algebras to Iterative Theories
[AMV_3]	Elgot Algebras
[AAMV]	Infinite Trees and Completely Iterative Theories: A Coalgebraic View
[M_1]	Completely Iterative Algebras and Completely Iterative Monads
[M_2]	On Iteratable Endofunctors
[MM]	The Category Theoretic Solution of Recursive Program Schemes

These papers contain the results with full proofs that we have presented in the preceeding sections. Each of the papers may, of course, be read on its own. Together they can serve as a reference for all the details we had to omit.

Math. Struct. in Comp. Science (2003), *vol.* 13, *pp.* 259–320.
DOI: 10.1017/S0960129502003924 Printed in the United Kingdom

Free iterative theories: a coalgebraic view

JIŘÍ ADÁMEK†, STEFAN MILIUS and JIŘÍ VELEBIL†

Institute of Theoretical Computer Science, Technical University, Braunschweig, Germany
Email: {adamek,milius,velebil}@iti.cs.tu-bs.de

Received 30 August 2001; revised 15 March 2002

Every finitary endofunctor of Set is proved to generate a free iterative theory in the sense of Elgot. This work is based on coalgebras, specifically on parametric corecursion, and the proof is presented for categories more general than just Set.

1. Introduction

Iterative algebraic theories were introduced by Calvin C. Elgot in Elgot (1975) as a concept serving the study of computation (on, say, Turing machines) at a level abstracting from the nature of external memory. The main example presented by Elgot is the theory of rational trees, that is, infinite trees that are solutions of systems of finitary iterative equations. Or, equivalently, that possess only finitely many subtrees. He and his coauthors later proved that this theory is a free iterative theory on a given (finitary) signature (Elgot *et al.* 1978).

The purpose of the present paper is to generalise Elgot's result from signatures (in other words, polynomial endofunctors of the category of sets) to finitary endofunctors of Set and some 'set-like' categories, for example, the category of posets. Using a very general Solution Theorem, developed in previous work, which shows by coalgebraic methods how iterative equations can be solved in categories, we prove that finitary endofunctors generate free iterative theories (in other words, finitary monads), called *rational monads.* We construct the rational monad in two steps:

(1) A rational monad of a *strongly finitary* endofunctor, that is, a finitary endofunctor 'preserving finiteness', is constructed in Section 4.

and

(2) A rational monad of a general finitary endofunctor is derived in Section 5.

We work with categories called strongly LFP, which we introduce in Section 2. And we assume that the given endofunctor preserves monomorphisms. For the category Set this last assumption can be omitted, as we show in Section 6. For all these cases the common formulation of a rational monad, R, is: R assigns to every object Y (of 'parameters') the union RY of all images of solutions of finitary flat equations with parameter Y.

In the rest of the introduction we explain the concepts mentioned above, and give further references.

† The first and third authors acknowledge the support of the Grant Agency of the Czech Republic under the Grant No. 201/02/0148.

1.1. *What is a rational tree?*

The algebra of finite and infinite Σ-labelled trees has, for every finitary signature Σ, the important property that iterative equational systems of a certain (liberal) type have unique solutions. And when we solve finitary flat iterative systems, we obtain precisely the so-called rational trees. We will now describe this in more detail.

For any set X of variables, we use

$$T_\Sigma X$$

to denote the algebra of all finite and infinite Σ-labelled trees with variables from X. That is, trees labelled so that a node with $n > 0$ children is labelled by an n-ary operation symbol (an element of Σ_n), and a leaf is labelled by a variable or a constant symbol (an element of $X + \Sigma_0$). The operations on $T_\Sigma X$ are given by tree-tupling. Now consider a system of iterative equations[†]

$$\begin{aligned} x_0 &\approx t_0(x_0, x_1, x_2, \ldots, y_0, y_1, y_2, \ldots) \\ x_1 &\approx t_1(x_0, x_1, x_2, \ldots, y_0, y_1, y_2, \ldots) \\ &\vdots \\ x_n &\approx t_n(x_0, x_1, x_2, \ldots, y_0, y_1, y_2, \ldots) \\ &\vdots \end{aligned} \tag{1}$$

where t_i are trees with leaves labelled by variables from $X = \{x_0, x_1, x_2, \ldots\}$ and parameters from $Y = \{y_0, y_1, y_2, \ldots\}$, that is,

$$t_i \in T_\Sigma(X + Y) \quad \text{for } i = 0, 1, 2, \ldots$$

Such a system is called *guarded* if none of the trees t_i is a variable from X. This condition alone guarantees that there exists a unique *solution* of (1), that is, a unique list ${x_i}^\dagger(y_0, y_1, y_2, \ldots)$ of trees in $T_\Sigma Y$ such that the expected identities

$$\begin{aligned} {x_0}^\dagger &= t_0({x_0}^\dagger, {x_1}^\dagger, {x_2}^\dagger, \ldots, y_0, y_1, y_2, \ldots) \\ {x_1}^\dagger &= t_1({x_0}^\dagger, {x_1}^\dagger, {x_2}^\dagger, \ldots, y_0, y_1, y_2, \ldots) \\ &\vdots \\ {x_n}^\dagger &= t_n({x_0}^\dagger, {x_1}^\dagger, {x_2}^\dagger, \ldots, y_0, y_1, y_2, \ldots) \\ &\vdots \end{aligned}$$

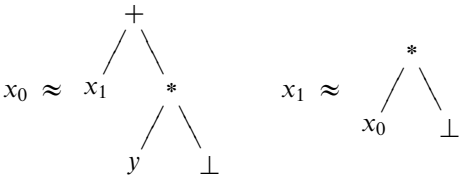
hold in $T_\Sigma Y$.

Theorem 1.1. Every guarded system of iterative equations has a unique solution.

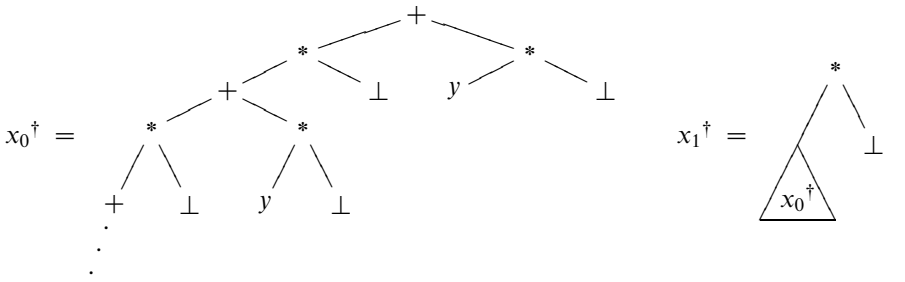
This is a special case of a much more general Solution Theorem, which we mention in Section 1.2 below.

[†] We use $\approx$ to denote formal equations and $=$ to denote the identity of the two sides.

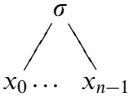
Example 1.2. Let Σ consist of binary operation symbols $+$ and $*$ and a constant symbol $\perp$. The following system of iterative equations



is guarded. The solution is given by the following trees in $T_\Sigma Y$:



A guarded equational system (1) is called *finitary* if it has only finitely many variables and the right-hand sides t_i are finite trees (such as those in Example 1.2). It is easy to see that every finitary system has a simple reduction to a finitary *flat* system, that is, one with finitely many variables, and with right-hand sides t_i that are either

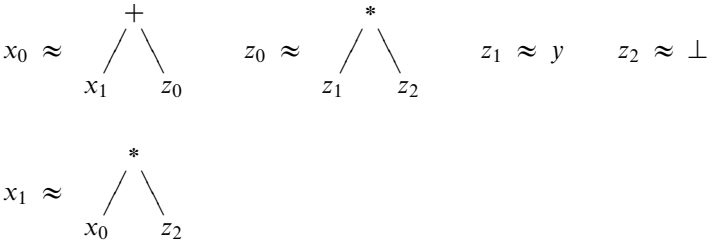
(a) flat trees  with $\sigma \in \Sigma_n$ and $x_0, \ldots, x_{n-1} \in X$

or

(b) single parameters from Y.

Observe that (a) includes single constant symbols $\sigma \in \Sigma_0$.

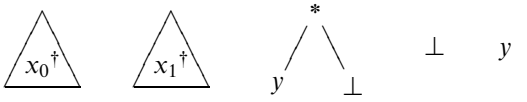
Example 1.3. A reduction of the system of Example 1.2 is obtained by introducing new variables z_0, z_1 and z_2 as follows:



The unique solution of this reduced system is the original solution $x_0{}^\dagger$ and $x_1{}^\dagger$ together with the obvious new trees $z_i{}^\dagger$ for $i = 0, 1, 2$.

Definition 1.4. A tree in $T_\Sigma Y$ is called *rational* if it can be obtained by solving a finitary flat system of equations.

Thus, the trees $x_0{}^\dagger$ and $x_1{}^\dagger$ in Example 1.2 are examples of rational trees. Rational trees are fully characterised as those trees in $T_\Sigma Y$ that have, up to tree isomorphism, only finitely many subtrees (Elgot *et al.* 1978). For example, all subtrees of $x_0{}^\dagger$ in Example 1.2 are isomorphic to one of the following five (obtained by breadth-first search of the nodes of $x_0{}^\dagger$):



Now the subalgebra

$$R_\Sigma Y$$

of $T_\Sigma Y$ of all rational trees has a solution property 'almost as strong' as the algebra $T_\Sigma Y$ itself.

Theorem 1.5. Every finite guarded system of iterative equations with rational right-hand sides has a unique solution in $R_\Sigma Y$.

A direct proof of this theorem is not difficult, but we do not have to discuss it here because we are going to prove a much more general result, called the Rational Solution Theorem, see Corollary 5.8.

1.2. *What is a solution in general?*

In this subsection we recall briefly some results, which were obtained independently by Larry Moss (Moss 2001) and the present authors in collaboration with Peter Aczel (Aczel *et al.* 2002). The reader can find more details in the extended abstract Aczel *et al.* (2001), which has already been published.

We now generalise infinite trees, which are elements of the final coalgebra $T_\Sigma X$ of the polynomial endofunctor $H_\Sigma(\_) + X$, to final coalgebras of $H(\_) + X$ for an arbitrary endofunctor H. Recall that, given a signature Σ, the corresponding *polynomial functor*

$$H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$$

defined by

$$H_\Sigma A = \Sigma_0 + \Sigma_1 \times A + \Sigma_2 \times A^2 + \cdots$$

has the property that H_Σ-algebras are just the classical universal algebras of signature Σ. A final H_Σ-coalgebra is well known to be the coalgebra $T_\Sigma \emptyset$ of all finite and infinite Σ-labelled trees without variables (Adámek and Koubek 1995). Now the functor $H_\Sigma(\_) + X$ is also polynomial (for the signature obtained from Σ by adding a constant symbol for every variable in X), thus,

$$T_\Sigma X \text{ is a final coalgebra of } H_\Sigma(\_) + X.$$

In our previous work (Aczel *et al.* 2002) we have introduced the concept of an *iteratable* endofunctor of Set (or, more generally, of any category with binary coproducts): it is an

endofunctor H such that $H(\_)+X$ has a final coalgebra, for every X. We use the notation

$$TX$$

to denote a final coalgebra of $H(\_)+X$, and the coalgebra-structure (which, by Lambek's Lemma (Lambek 1968), is an isomorphism between TX and $HTX+X$) is denoted by giving names to the coproduct injections of TX (as a coproduct of HTX and X) as follows:

$$\eta_X : X \longrightarrow TX \quad \text{('injection of variables')}$$

and

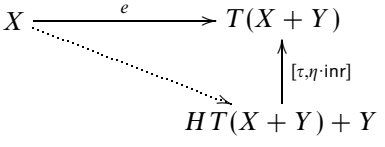
$$\tau_X : HTX \longrightarrow TX \quad (\text{'}TX \text{ becomes an } H\text{-algebra'}).$$

Recall that an endofunctor is called *finitary* if it preserves filtered colimits. Every finitary endofunctor is iteratable, see Example 2.11 of Aczel *et al.* (2002). The above concept of a guarded system of equations can be formalised by a function $e : X \longrightarrow T(X+Y)$, $x_i \mapsto t_i$. And since $T(X+Y) = \big(HT(X+Y)+Y\big)+X$ is a coproduct with injections $[\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}] : HT(X+Y)+Y \longrightarrow T(X+Y)$ and $\eta_{X+Y} \cdot \mathsf{inl} : X \longrightarrow T(X+Y)$, to say that the right-hand sides t_i are never variables from X is precisely to say that e factors through $[\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}]$. Hence, the notion of a guarded system (1) generalises as follows.

Definition 1.6. Let H be an iteratable endofunctor. We define a *guarded equation morphism* for H to be a morphism of the form

$$e : X \longrightarrow T(X+Y)$$

(X is the 'object of variables' and Y the 'object of parameters') that factors through the coproduct injection $[\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}] : HT(X+Y)+Y \longrightarrow T(X+Y)$:



The H-algebras TX have a rich structure. First, substitution of trees in $T_\Sigma X$ for variables generalises to all iteratable endofunctors. Recall that given an interpretation of variables $x \in X$ as trees $s(x)$ over Y, that is, a function $s : X \longrightarrow T_\Sigma Y$, the corresponding substitution of trees from $T_\Sigma Y$ into (leaves of) trees of $T_\Sigma X$ is a homomorphism

$$\widehat{s} : T_\Sigma X \longrightarrow T_\Sigma Y$$

of Σ-algebras. Moreover, $\widehat{s}$ is the unique homomorphic extension of s. This can be generalised to all iteratable endofunctors.

Theorem 1.7 (Substitution Theorem). For every morphism $s : X \longrightarrow TY$ there exists a unique homomorphism $\widehat{s} : TX \longrightarrow TY$ of H-algebras extending s (that is, with $s = \widehat{s}\,\eta_X$).

The proof can be found in Moss (2001) or Aczel *et al.* (2001) (and a slightly improved version in Aczel *et al.* (2002)).

Corollary 1.8. $(T, \eta, \widehat{(\_)})$ is a *Kleisli triple*, that is, the following three axioms are satisfied:

(i) $\widehat{\eta_X} = id_{TX}$, for every object X,
(ii) $\widehat{s}\,\eta_X = s$, for every morphism $s : X \longrightarrow TY$,
and
(iii) $\widehat{r}\,\widehat{s} = \widehat{\widehat{r}\,s}$, for every pair $s : X \longrightarrow TY$ and $r : Y \longrightarrow TZ$.

As shown in Manes (1976), axioms (i)–(iii) are equivalent to having a monad (T, η, μ), where $\mu_X = \mathrm{i}\widehat{d_{TX}} : TTX \longrightarrow TX$. Observe that μ_X is a homomorphism of H-algebras (since every $\widehat{s}$ is). However, this monad is usually not finitary; thus, it cannot be identified with an algebraic theory in the sense of Lawvere.

Example 1.9. For every signature Σ we have a monad (T_Σ, η, μ), where $T_\Sigma X$ is the algebra of all finite and infinite Σ-labelled trees above, η is the insertion of variables (as one-node trees), and μ is the usual substitution of trees into trees. This monad is seldom finitary. (In fact, it is finitary iff all operations of Σ are either unary or nullary.)

Next, we can introduce solutions for equation morphisms $e : X \longrightarrow T(X + Y)$ by mimicking the case of trees as follows. A solution of e is a morphism $e^\dagger : X \longrightarrow TY$. It has the property that when the following 'substitution' morphism

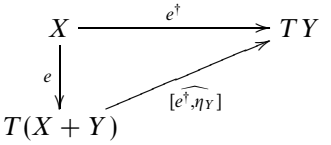
$$s = [e^\dagger, \eta_Y] : X + Y \longrightarrow TY$$

is considered (that is, every variable x_i is substituted by the tree $e^\dagger(x_i) = {x_i}^\dagger \in TY$, while parameters are unchanged), the composite

$$X \xrightarrow{\ e\ } T(X + Y) \xrightarrow{\ \widehat{s}\ } TY$$

(corresponding to performing the substitution s on all variables of the right-hand sides of (1)) is equal to $e^\dagger$.

Definition 1.10. We define a *solution* of an equation morphism $e : X \longrightarrow T(X + Y)$ to be a morphism $e^\dagger : X \longrightarrow TY$ such that the triangle



commutes.

The following result is called Parametric Corecursion in Moss (2001) and Solution Theorem in Aczel *et al.* (2001); see also a much improved version of the proof in Aczel *et al.* (2002).

Theorem 1.11 (Solution Theorem). Given an iteratable endofunctor H, every guarded equation morphism has a unique solution.

1.3. *What is a rational monad?*

Consider a finitary endofunctor H of Set (that is, H preserves filtered colimits). The following generalises the above finitary flat systems of equations for $H = H_\Sigma$.

Definition 1.12. We define a *finitary flat equation morphism* for H to be a morphism of the following form

$$e : X \longrightarrow HX + Y \quad (X \text{ finite}).$$

In brief, finitary flat equation morphisms are just the finite coalgebras of the endofunctor $H(\_) + Y$. (If $H = H_\Sigma$, then a finitary flat equation morphism

$$e : X \longrightarrow \coprod_{n\in\omega} \Sigma_n \times X^n + Y \quad (X \text{ finite})$$

is precisely the concept as introduced above.) Every finitary flat equation morphism gives rise to a guarded equation morphism using $\tau_X : HTX \longrightarrow TX$ above as follows:

$$X \xrightarrow{\ e\ } HX + Y \xrightarrow{\ \tau_X H\eta_X + Y\ } TX + Y \xrightarrow{\ [T\mathsf{inl}, \eta_{X+Y}\cdot \mathsf{inr}]\ } T(X + Y). \tag{2}$$

By a harmless abuse of notation, we use $e^\dagger : X \longrightarrow TY$ to denote the unique solution of (2). We have proved in Aczel *et al.* (2002) that for flat equation morphisms we have

$$\text{solution} = \text{corecursion}.$$

That is, $e^\dagger$ is the unique homomorphism of the coalgebra X into the final coalgebra TY.

Mimicking the above definition of $R_\Sigma Y$ as the algebra of all solutions of finitary flat equation systems (for $H = H_\Sigma$), we introduce below an H-algebra

$$RY = \bigcup \mathrm{im}(e^\dagger)$$

as a union of images of all solutions of finitary flat equation morphisms with the parameter set Y (and an arbitrary finite set X of variables). In Adámek *et al.* (2002) we show that this definition of RY can be formulated equivalently using all finitary equation morphisms, not just the flat ones.

These H-algebras RY, which are subalgebras of TY, also have a rich structure; we are going to prove the following.

Rational Substitution Theorem: Every morphism $s : X \longrightarrow RY$ has a unique extension into a homomorphism $\widetilde{s} : RX \longrightarrow RY$ of H-algebras (see Theorem 5.14).

Rational Solution Theorem: Every guarded equation morphism $e : X \longrightarrow R(X + Y)$ where X is finite has a unique solution $e^\ddagger : X \longrightarrow RY$ (see Corollary 5.8).

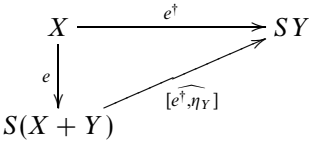
As a corollary of the Rational Substitution Theorem, we conclude immediately that R is also a monad on Set, this time a finitary monad (or, equivalently, Lawvere's algebraic theory (Manes 1976)). We call this monad the *rational monad* generated by the given finitary endofunctor H.

1.4. *A rational monad is a free iterative monad*

C. C. Elgot has studied algebraic theories (in other words, finitary monads on Set) with the property that certain iterative equations have unique solutions; he called such theories iterative (Elgot 1975). From Section 1.2 above, it is quite natural to call, for a given monad (S, η, μ), any morphism

$$e : X \longrightarrow S(X + Y)$$

an *equation morphism* (for S). Recall that every monad defines substitution: given $s : X \longrightarrow SY$ we have the corresponding homomorphism $\widehat{s} = \mu_Y \cdot Ss : SX \longrightarrow SY$ of free S-algebras. Now, a morphism $e^\dagger : X \longrightarrow SY$ will be called a *solution* of e if the triangle



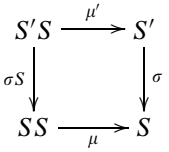
commutes. But how can we express the property of e being guarded? (Or *ideal*, which is a slightly stricter concept that Elgot used: for trees this means that the right-hand sides are neither variables, nor single parameters. But this difference is inessential as we prove below – see Remark 4.31.) For that purpose, Elgot introduced the concept of an ideal algebraic theory. Translated into the language of monads, it yields the following definition (see Aczel *et al.* (2002) for a very simple proof that our formulation is equivalent to Elgot's):

Definition 1.13. A monad (S, η, μ) is called *ideal* if:

(i) $S = S' + Id$ and $\eta : Id \longrightarrow S$ is the right-hand injection (notation: $\sigma : S' \longrightarrow S$ for the left-hand one)

and

(ii) μ restricts to $\mu' : S'S \longrightarrow S'$, that is, there is a natural transformation μ' such that the square

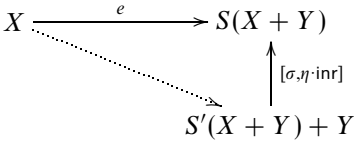


commutes.

This definition applies to all categories with finite coproducts such that coproduct injections are monomorphic.

Example 1.14. The monad T defined by Corollary 1.8 is ideal: here $T = HT + Id$ with injections τ (=σ) and η. Also, the monad R defined by the Rational Substitution Theorem is ideal, as we prove below.

For ideal monads (S, η, μ) we call an equation morphism $e : X \longrightarrow S(X + Y)$ *guarded* if it factors through $[\sigma_{X+Y}, \eta_{X+Y}\,\mathsf{inr}] : S'(X + Y) + Y \longrightarrow S(X + Y)$:



Definition 1.15. (Elgot 1975) A finitary monad S in Set is called *iterative* if it is ideal and every guarded equation morphism $e : X \longrightarrow S(X + Y)$ with X and Y finite has a unique solution.

The restriction to finite sets Y, in comparison to the Rational Solution Theorem above, is inessential because the rational monad is finitary. Thus, results using finitely presentable parameter objects Y extend easily to results on arbitrary Y.

The main result of the present paper is the fact that every finitary endofunctor of Set generates a free iterative monad, *viz*., the rational monad. For polynomial endofunctors this is the main result of Elgot *et al.* (1978), but is proved in a completely different manner.

This situation is analogous to the main result in Aczel *et al.* (2002) concerning *completely iterative monads*, as introduced in Elgot *et al.* (1978): they are defined as above, except that the requirement that X and Y be finite is dropped. We have proved that every iteratable endofunctor generates a free completely iterative monad, *viz*., the above monad T.

1.5. *Beyond the category of sets*

The concept of an ideal monad is immediately extended to any category with finite coproducts. And we can also extend Definition 1.15 naturally, as follows.

Definition 1.16. A monad S is called *iterative* if it is ideal and every guarded equation morphism $e : X \longrightarrow S(X + Y)$ with X and Y finitely presentable (see Section 2.2 below) has a unique solution.

However, for the main result of our paper, stating that every finitary endofunctor generates a free iterative monad, we need to make rather strong additional assumptions on the category we work with. This contrasts to the situation of Aczel *et al.* (2002), where the result that every iteratable functor generates a free completely iterative monad is proved for all categories having finite coproducts with monomorphic injections. In the present paper we will assume that the given category is strongly LFP, that is, it has to be extensive and locally finitely presentable with finitely presentable objects closed under strong quotients and have finite hom-sets for finitely presentable objects. We introduce these concepts in Section 2.

Examples 1.17. The category Set^S (for S-sorted operations) is strongly LFP. Also, the category of posets (for order-enriched operations) is strongly LFP.

The only additional assumption on the finitary endofunctor is that it preserves monomorphisms, an assumption not needed for set functors, as we show in Section 6.

In the next section we recall everything about locally finitely presentable categories that we need later. Even for readers only interested in $\mathscr{A} = \mathsf{Set}$, this section brings information about finitary monads that will be used further.

2. Finitary monads

2.1. *Finitary Kleisli triples in* Set.

Recall the concept of a *Kleisli triple* $(T, \eta, \widehat{(\_)})$ on a category $\mathscr{A}$, which is equivalent to that of a monad on $\mathscr{A}$ (Manes 1976). It consists of:

(1) A function assigning to every object X in $\mathscr{A}$ an object TX and a morphism $\eta_X : X \longrightarrow TX$.

and

(2) A function assigning to every morphism $s : X \longrightarrow TY$ a morphism $\widehat{s} : TX \longrightarrow TY$ so that the axioms (i)–(iii) of Corollary 1.8, hold.

A *finitary Kleisli triple in* Set consists, analogously, of two functions: one assigns to every finite set X a set TX and a morphism $\eta_X : X \longrightarrow TX$, and the other one assigns to every morphism $s : X \longrightarrow TY$ with X and Y finite a morphism $\widehat{s} : TX \longrightarrow TY$ so that (i)–(iii) of corollary hold. In Adámek *et al.* (2002) we proved that there exists a unique *finitary monad* on Set, that is, a monad (T, η, μ) such that T preserves filtered colimits, generating that triple in the expected sense:

(a) For every finite set X, the given $\eta_X : X \longrightarrow TX$ is the component of the unit $\eta : Id \longrightarrow T$.

and

(b) For every $s : X \longrightarrow TY$ with X and Y finite, the usual 'substitution formula'

$$\widehat{s} = \mu_Y \cdot Ts : TX \longrightarrow TY$$

holds.

In other words, a finitary Kleisli triple can be extended to a Kleisli triple corresponding to a finitary monad, and vice versa.

Let us also recall from Manes (1976) that finitary monads on Set are precisely the same as Lawvere's algebraic theories: given a finitary monad (T, η, μ), we can consider its theory consisting of natural numbers as objects, and functions $n \longrightarrow Tk$ as morphisms from n to k. The composite of

$$n \xrightarrow{f} Tk \quad \text{and} \quad k \xrightarrow{g} Tl$$

is

$$n \xrightarrow{f} Tk \xrightarrow{Tg} TTl \xrightarrow{\mu_l} Tl$$

Conversely, every Lawvere's theory represents a finitary variety $\mathscr{V}$ whose monad (describing $\mathscr{V}$-free algebras) is finitary.

2.2. *Locally finitely presentable (LFP) categories*

We generalise the concept of a finite set in Set to that of a *finitely presentable object* A in a category $\mathscr{A}$: it is an object such that the hom-functor $\mathscr{A}(A, \_) : \mathscr{A} \longrightarrow \mathsf{Set}$ preserves filtered colimits.

Definition 2.1. (Gabriel and Ulmer 1971). A category is called LFP if it is cocomplete and has a set of finitely presentable objects whose closure under filtered colimits is the whole category.

Remark 2.2. It follows that every LFP category $\mathscr{A}$ has, up to isomorphism, only a set of finitely presentable objects. Moreover, if

$$\mathscr{A}_{fp}$$

denotes a set of representatives of finitely presentable objects (considered as a full subcategory), then for every object X of $\mathscr{A}$ we have:

(i) The comma-category $\mathscr{A}_{fp}/X$ (consisting of all morphisms $a : A \longrightarrow X$ with A in $\mathscr{A}_{fp}$) is filtered.

and

(ii) X is a canonical colimit of the diagram

$$D_X : \mathscr{A}_{fp}/X \longrightarrow \mathscr{A}$$

given by

$$D_X(a : A \longrightarrow X) = A.$$

Consequently,

(iii) $\mathscr{A}$ is a free cocompletion of $\mathscr{A}_{fp}$ under filtered colimits. That is, every functor $F : \mathscr{A}_{fp} \longrightarrow \mathscr{B}$, where $\mathscr{B}$ has filtered colimits, has an extension, unique up to natural isomorphism, to a functor $F' : \mathscr{A} \longrightarrow \mathscr{B}$ preserving filtered colimits.

See Adámek and Rosický (1994).

Examples 2.3.

(i) Set is LFP, choose Set_{fp} to be the full subcategory of natural numbers.
(ii) The category Pos of posets and monotone maps is LFP, here Pos_{fp} is a full subcategory representing all finite (in other words, finitely presentable) posets.
(iii) The category Gra of graphs (sets with a binary relation) and graph homomorphisms is LFP with Gra_{fp} representing all finite graphs.
(iv) Every variety $\mathscr{V}$ of (finitary, many-sorted) algebras is LFP. An algebra V in $\mathscr{V}$ is a finitely presentable object iff it can be presented by finitely many generators and finitely many equations.

Definition 2.4. We define a *finitary Kleisli triple* on an LFP category $\mathscr{A}$ to be a triple $(T, \eta, \widehat{(\_)})$ where T is a function assigning to every finitely presentable object X an object TX, with a morphism $\eta_X : X \longrightarrow TX$, and $\widehat{(\_)}$ is a function assigning to every morphism

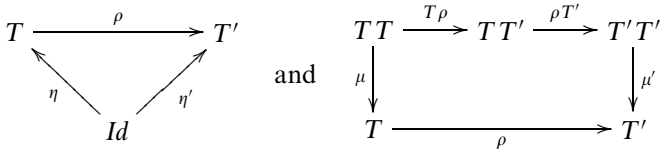
$s : X \longrightarrow TY$ with X and Y finitely presentable a morphism $\widehat{s} : TX \longrightarrow TY$ so that:

(i) $\widehat{\eta_X} = id_{TX}$, for all X in $\mathscr{A}_{fp}$;
(ii) $\widehat{s}\,\eta_X = s$, for all $s : X \longrightarrow TY$, (X, Y in $\mathscr{A}_{fp}$);
and
(iii) $\widehat{r}\,\widehat{s} = \widehat{\widehat{r}\,s}$, for all $s : X \longrightarrow TY$ and $r : Y \longrightarrow TZ$ (X, Y, Z in $\mathscr{A}_{fp}$).

Example 2.5. Every finitary monad (T, η, μ) on an LFP category $\mathscr{A}$ generates a finitary Kleisli triple with $\widehat{s} = \mu_Y \cdot Ts$ for all $s : X \longrightarrow TY$ with X, Y in $\mathscr{A}_{fp}$.

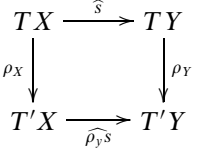
Remark 2.6. It is proved in Adámek *et al.* (2002) that finitary Kleisli triples on an LFP category $\mathscr{A}$ correspond precisely to finitary monads on $\mathscr{A}$. In fact, the correspondence extends to the level of morphisms of Kleisli triples and monads.

Recall that, given monads (T, η, μ) and (T', η', μ') on a category, a *monad homomorphism* is a natural transformation $\rho : T \longrightarrow T'$ such that the diagrams



commute. If the components of ρ are monomorphisms, then (T, η, μ) is called a *submonad* of (T', η', μ').

Expressed by Kleisli triples, a homomorphism is given by assigning, to every object X, a morphism $\rho_X : TX \longrightarrow T'X$ such that (a) $\rho_X \cdot \eta_X = \eta'_X$ and, (b) given $s : X \longrightarrow TY$, the square



commutes (where we use $\widehat{(\_)}$ for both Kleisli triples). When both triples are finitary, we restrict to X and Y finitely presentable in the above definition.

The precise formulation of the correspondence mentioned above is given by the following proposition (Adámek *et al.* 2002).

Proposition 2.7. Let $\mathscr{A}$ be an LFP category. The category of finitary Kleisli triples on $\mathscr{A}$ and their homomorphisms is equivalent to the category of finitary monads on $\mathscr{A}$ and their homomorphisms.

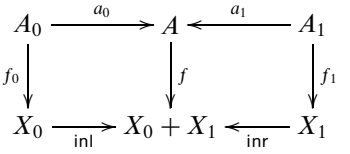
Corollary 2.8. The category of finitary monads on $\mathscr{A}$ is coreflective in the category of all monads on $\mathscr{A}$. A coreflection of a monad T is the finitary monad generating the same finitary Kleisli triple as T.

Remark 2.9. From Corollary 2.8 it follows trivially that for a finitary monad R and an arbitrary monad T, given any morphism of the corresponding finitary Kleisli triples, there is a unique extension to a monad homomorphism from R to T.

2.3. *Strongly LFP categories*

In the subsequent sections we have several additional requirements on the LFP category we work with, which we now summarise in an (admittedly *ad hoc*) definition. The main point is that Set has all these properties.

Recall that a category with coproducts and pullbacks is said to be *extensive* if every commutative diagram



comprises a pair of pullback squares iff the top row is a coproduct diagram. Equivalently, coproducts are disjoint and universal (Carboni *et al.* 1993).

In particular, extensive categories have the property that

coproduct injections are monomorphisms,

and

a coproduct of two monomorphisms is monomorphic.

Recall that an epimorphism e is called *strong* iff it has the diagonal fill-in property with respect to all monomorphisms m (that is, given morphisms u, v with $ve = mu$, there exists d with $u = de$ and $v = md$). *Strong quotients* are quotients carried by strong epimorphisms.

Remark 2.10. Every LFP category has a factorisation of morphisms as strong epimorphisms followed by monomorphisms (Adámek and Rosický 1994, 1.61).

Definition 2.11. A category is called *strongly LFP* if it is LFP and extensive, hom-sets of finitely presentable objects are finite, and a strong quotient of a finitely presentable object is finitely presentable.

Examples 2.12.

(i) Set is strongly LFP.
(ii) Pos and Gra are strongly LFP.
(iii) Every category of relational structures (of any finitary relational signature) is strongly LFP: the finitely presentable objects are those with finitely many elements, and with all relations but finitely many empty.

2.4. *Finitary endofunctors of an LFP category*

It follows from Remark 2.2(iii) above that the category $\mathsf{Fin}[\mathscr{A}, \mathscr{A}]$ of finitary endofunctors of an LFP category $\mathscr{A}$ is equivalent to the functor category $[\mathscr{A}_{fp}, \mathscr{A}]$. From this result it follows that $\mathsf{Fin}[\mathscr{A}, \mathscr{A}]$ is itself an LFP category. We now describe its finitely presentable objects.

For any pair A, B of finitely presentable objects of on LFP category $\mathscr{A}$ we use $A \bullet B : \mathscr{A}_{fp} \longrightarrow \mathscr{A}$ to denote the following functor:

$$A \bullet B : X \mapsto \coprod_{\mathscr{A}_{fp}(A,X)} B. \tag{3}$$

These functors were called *step functors* in Adámek (1997) and were proved to be finitely presentable in $[\mathscr{A}_{fp}, \mathscr{A}]$ (see Lemma 1 there).

Proposition 2.13. Every functor F in $[\mathscr{A}_{fp}, \mathscr{A}]$ is a filtered colimit of functors that are finite colimits of step functors.

Proof. See the proof of Theorem 3 in Adámek (1997). □

Definition 2.14. A finitary functor $F : \mathscr{A} \longrightarrow \mathscr{A}$ is called *strongly finitary* if it preserves finitely presentable objects.

Proposition 2.15. For an LFP category $\mathscr{A}$ the following are equivalent:

(i) The category $\mathscr{A}_{fp}$ has finite hom-sets.
(ii) Every finitary endofunctor of $\mathscr{A}$ is a filtered colimit of strongly finitary endofunctors.

Proof. (i) $\Rightarrow$ (ii): Since the category of finitary endofunctors of $\mathscr{A}$ is equivalent to $[\mathscr{A}_{fp}, \mathscr{A}]$, it suffices to show that every functor $F : \mathscr{A}_{fp} \longrightarrow \mathscr{A}$ is a filtered colimit of functors that preserve finitely presentable objects.

By (i), every functor $A \bullet B$ is strongly finitary and a finite colimit of strongly finitary functors is strongly finitary. Now use Proposition 2.13.

(ii) $\Rightarrow$ (i): Suppose that A and B are finitely presentable objects in $\mathscr{A}$ with $\mathscr{A}_{fp}(A, B)$ infinite. We show that the object

$$(A \bullet A)B = \coprod_{\mathscr{A}_{fp}(A,B)} A \tag{4}$$

is not finitely presentable in $\mathscr{A}$, and therefore the functor $A \bullet A$ is not strongly finitary. Since $A \bullet A$ is finitely presentable, it cannot be expressed as a filtered colimit of strongly finitary functors: in fact, $A \bullet A$ would be a retract of one of them, but retracts of strongly finitary functors are strongly finitary.

We prove that $\coprod_J A$ is not finitely presentable for any infinite set J. In fact, $\coprod_J A$ is a filtered colimit of $\coprod_I A$ for all finite sets $I \subseteq J$. If $\coprod_J A$ were finitely presentable, some of the colimit maps, that is, some canonical map

$$c_I : \coprod_I A \longrightarrow \coprod_J A \qquad (I \text{ finite})$$

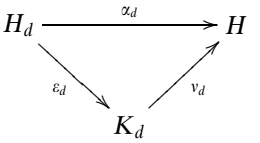
would be a split epimorphism (because the identity of $\coprod_J A$ would factor through c_I, by definition of finite presentability). However, c_I is not an epimorphism: choose distinct morphisms $f, g \in \mathscr{A}_{fp}(A, B)$ and let $\overline{f}, \overline{g} : \coprod_J A \longrightarrow B$ be morphisms whose components are equal for all indices in I, but whose j-components (for some $j \in J \setminus I$) are f and g, respectively. Then $\overline{f} \neq \overline{g}$ but $\overline{f} c_I = \overline{g} c_I$. □

Corollary 2.16. Suppose that $\mathscr{A}$ is a strongly LFP category. Every finitary endofunctor H preserving monomorphisms can be expressed as a filtered colimit of strongly finitary functors preserving monomorphisms.

Proof. Use Proposition 2.15 and express H as a filtered colimit

$$H = \operatorname*{colim}_d H_d$$

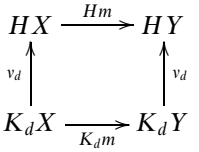
of strongly finitary functors and use $\alpha_d : H_d \longrightarrow H$ to denote a colimit cocone. The category $\mathsf{Fin}[\mathscr{A}, \mathscr{A}] \simeq [\mathscr{A}_{fp}, \mathscr{A}]$ of finitary endofunctors of $\mathscr{A}$ is LFP, therefore, it has (StrongEpi, Mono)-factorisations of morphisms, see Remark 2.10. Moreover, both strong epimorphisms and monomorphisms are defined component-wise in any functor category. We can factor each $\alpha_d : H_d \longrightarrow H$ in $\mathsf{Fin}[\mathscr{A}, \mathscr{A}]$ as a strong epimorphism followed by a monomorphism:



Since each ε_d is pointwise a strong epimorphism, every functor K_d is strongly finitary: given a finitely presentable object X, we have K_dX is a strong quotient of a finitely presentable object H_dX (we know that H_d is strongly finitary), thus, K_dX is finitely presentable.

It is easy to see that the cocone $(v_d : K_d \longrightarrow H)$ is a filtered colimit, this follows, once again from the fact that $\varepsilon_d : H_d \longrightarrow K_d$ are (strong) epimorphisms.

It remains to show that every functor K_d preserves monomorphisms. Let $m : X \longrightarrow Y$ be a monomorphism. From commutativity of the square



it follows that $(v_d)_Y \cdot K_dm$ is a monomorphism, thus, K_dm is a monomorphism. □

3. Some properties of solutions

Assumptions 3.1. We use H to denote a finitary, monomorphism preserving endofunctor of a strongly LFP category $\mathscr{A}$.

Notice that every finitary functor is iteratable, which implies (as in the case $\mathscr{A} = \mathsf{Set}$ mentioned in Section 1.2 above) that final coalgebras

$$TX$$

of $H(\_) + X$ form a 'completely iterative' monad T (Aczel *et al.* 2001). As in Section 1.2, T is a coproduct $T = HT + Id$ with injections

$$\tau : HT \longrightarrow T \quad \text{and} \quad \eta : Id \longrightarrow T.$$

Notation 3.2. Put

$$\tau^* \equiv H \xrightarrow{H\eta} HT \xrightarrow{\tau} T.$$

Observe that τ, η and τ^* have monomorphic components (because $\mathscr{A}$ has monomorphic coproduct injections, and H preserves monomorphisms).

The concept of a flat system of equations as introduced in the Introduction (see Sections 1.1 and 1.3) immediately generalises as follows.

Definition 3.3. We define a *flat equation morphism* to be a morphism

$$e : X \longrightarrow HX + Y,$$

that is, a coalgebra of $H(\_) + Y$. It is called *finitary* if X is finitely presentable.

Remark 3.4. Every flat equation morphism $e : X \longrightarrow HX + Y$ yields a guarded equation morphism in the sense of Section 1.3 by composing with the following monomorphism

$$m_{X,Y} \equiv HX + Y \xrightarrow{\tau^* + Y} TX + Y \xrightarrow{[T\mathsf{inl}, \eta_{X+Y} \cdot \mathsf{inr}]} T(X + Y).$$

We also write, whenever there is no danger of confusion,

$$e^\dagger : X \longrightarrow TY$$

for the unique solution of the corresponding guarded equation. Thus $e^\dagger$ is the unique morphism such that the square

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & TY \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu_Y} \\ HX + Y & & \\ {\scriptstyle m_{X,Y}}\downarrow & & \\ T(X + Y) & \xrightarrow[T[e^\dagger, \eta]]{} & T^2 Y \end{array}$$

commutes. The following proposition was proved in Aczel *et al.* (2002).

Proposition 3.5 (Solution is Corecursion). For every flat equation morphism $e : X \longrightarrow HX + Y$, the (unique) solution $e^\dagger : X \longrightarrow TY$ is precisely the unique homomorphism of the coalgebra $e : X \longrightarrow HX + Y$ into the final coalgebra TY of $H(\_) + Y$.

Remark 3.6. We also need a simple result concerning general iterative monads, see Definitions 1.13 and 1.16.

Let S be an iterative monad. An equation morphism $e : X \longrightarrow S(X + Y)$ that factors through σ_{X+Y} is, of course, guarded. Moreover, the unique solution $e^\dagger : X \longrightarrow SY$ factors through σ_Y: if $e = \sigma_{X+Y} \cdot e'$, then

$$e^\dagger = \mu_Y \cdot S[e^\dagger, \eta_Y] \cdot \sigma_{X+Y} \cdot e' = \sigma_Y \cdot (\mu'_Y \cdot S'[e^\dagger, \eta_Y] \cdot e').$$

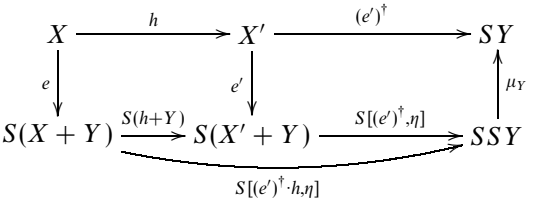
Lemma 3.7. Let S be an iterative monad and let $e : X \longrightarrow S(X + Y)$ and $e' : X' \longrightarrow S(X' + Y)$ be guarded equation morphisms. For every morphism $h : X \longrightarrow X'$ such that the square

$$\begin{array}{ccc} X & \xrightarrow{e} & S(X+Y) \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle S(h+Y)} \\ X' & \xrightarrow[e']{} & S(X'+Y) \end{array}$$

commutes, we have

$$e^\dagger = (e')^\dagger \cdot h.$$

Proof. The diagram



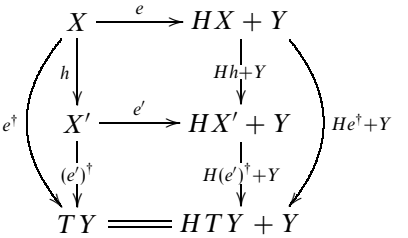
obviously commutes. □

Remark 3.8.

(i) Returning to flat equation morphisms, the corresponding statement here is a direct corollary of Proposition 3.5: given flat equation morphisms $e : X \longrightarrow HX + Y$ and $e' : X' \longrightarrow HX' + Y$, we have, for every homomorphism $h : X \longrightarrow X'$ of those coalgebras (of $H(\_) + Y$), that

$$e^\dagger = (e')^\dagger \cdot h.$$

(ii) In particular, when both $e^\dagger$ and $(e')^\dagger$ happen to be monomorphisms (that is, subobjects of TY), then the existence of a homomorphism $h : e \longrightarrow e'$ implies $e^\dagger \subseteq (e')^\dagger$. The converse is also true: if $e^\dagger \subseteq (e')^\dagger$, then the unique $h : X \longrightarrow X'$ with $e^\dagger = (e')^\dagger h$ is a homomorphism. In fact, consider the following diagram:



The outer and lower squares commute by Proposition 3.5. Hence the upper square commutes when extended by $H(e')^\dagger + Y$, which is a monomorphism since H preserves monomorphisms (by assumption) and monomorphisms are closed under coproducts. Thus $h : e \longrightarrow e'$ is a homomorphism, as desired.

4. Rational monad – strongly finitary case

Assumptions 4.1. Throughout this section, $\mathscr{A}$ denotes a strongly LFP category, and H an endofunctor that preserves monomorphisms and is *strongly finitary*, that is, not only preserves filtered colimits but also finite presentability (if X is a finitely presentable object, then so is HX).

In the next section we extend the results to all finitary endofunctors preserving monomorphisms.

Recall from Remark 2.10 that the category $\mathscr{A}$ has (StrongEpi, Mono)-factorisations of morphisms. We use $\mathrm{im}(f)$ to denote the monomorphic part of the factorisation of f.

The following definition generalises the definition of the algebra $R_\Sigma Y$ of all rational trees as solutions of systems of finitary flat equations, see Section 1.1.

Definition 4.2. For every finitely presentable object Y of $\mathscr{A}$, we define

$$RY = \bigcup \mathrm{im}(e^\dagger) \qquad (e \text{ a finitary flat equation morphism})$$

that is, we define an object RY together with a monomorphism

$$\varepsilon_Y : RY \longrightarrow TY$$

to be the subobject of TY that is the union of images of all solutions $e^\dagger : X \longrightarrow TY$ of finitary flat equation morphisms $e : X \longrightarrow HX + Y$ (X finitely presentable).

Remark 4.3. Explicitly, for every finitary flat equation morphism $e : X \longrightarrow HX + Y$, we factor $e^\dagger : X \longrightarrow TY$ as a strong epimorphism $k : X \longrightarrow X'$ followed by a monomorphism $m : X' \longrightarrow TY$. Then ε_Y is the union of all those monomorphisms m. Union here means just the supremum in the lattice of subobjects, but this is actually a colimit, as we see below.

Observe that each of the images m is, itself, a solution of a finitary flat equation morphism $e' : X' \longrightarrow HX' + Y$. In fact, since finitely presentable objects are closed under strong quotients, X' is finitely presentable, and, since Hm is a monomorphism (by assumption on H), so is $Hm + Y$ (since coproducts are extensive, and thus monomorphisms are stable under finite coproducts, see Section 2.3). Thus we can use the diagonal fill-in property to find a coalgebra structure e' on X' such that the diagram

$$\begin{array}{ccc} X & \xrightarrow{e} & HX + Y \\ {\scriptstyle k}\downarrow & & \downarrow{\scriptstyle Hk+Y} \\ X' & \xrightarrow{e'} & HX' + Y \\ {\scriptstyle m}\downarrow & & \downarrow{\scriptstyle Hm+Y} \\ TY & = & HTY + Y \end{array}$$

commutes. Now m is a coalgebra homomorphism, and it follows from Proposition 3.5 that $m = (e')^\dagger$.

Notation 4.4. For every finitely presentable object Y of $\mathscr{A}$, we use

$$\mathsf{EQ}_Y$$

to denote the category of all finitary flat equations with the parameter object Y. That is:

objects are all finitary flat equation morphisms $e : X \longrightarrow HX + Y$ (X in $\mathscr{A}_{fp}$)

and

morphisms are coalgebra homomorphisms with respect to $H(\_) + Y$:

$$\begin{array}{ccc} X & \xrightarrow{e} & HX + Y \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh+Y} \\ X' & \xrightarrow[e']{} & HX' + Y \end{array}$$

The category EQ_Y comes with a forgetful functor into the underlying category $\mathscr{A}$. We denote it by

$$\mathrm{Eq}_Y : \mathsf{EQ}_Y \longrightarrow \mathscr{A}, \qquad (X \xrightarrow{e} HX + Y) \mapsto X.$$

Remark 4.5. Recall that a full subcategory $\mathscr{D}_0$ of a filtered category $\mathscr{D}$ is *cofinal* if every object of $\mathscr{D}$ has a morphism into an object of $\mathscr{D}_0$; it follows that (in every category) colimits of $\mathscr{D}$-diagrams coincide with colimits of the restricted $\mathscr{D}_0$-diagrams.

Proposition 4.6. For every finitely presentable object Y, Eq_Y is a small, filtered diagram, and RY can be defined as a colimit

$$RY = \operatorname{colim} \mathrm{Eq}_Y.$$

Proof. First, notice that EQ_Y is the full subcategory of $\mathsf{Coalg}(H(\_) + Y)$ of those coalgebras with carrier from $\mathscr{A}_{fp}$. Since $\mathscr{A}_{fp}$ is small and for each object X of $\mathscr{A}$ there is only a set of morphisms $e : X \longrightarrow HX + Y$, it follows that EQ_Y is a small category. Moreover, observe that EQ_Y is filtered because it is finitely cocomplete: in fact, the category of coalgebras has colimits formed on the level of $\mathscr{A}$, and since finite colimits of finitely presentable objects are finitely presentable, it follows that EQ_Y is closed under finite colimits in $\mathsf{Coalg}(H(\_) + Y)$. Remark 4.3 shows that the filtered diagram Eq_Y has a cofinal subdiagram formed by all finitary flat equation morphisms with monomorphic solutions. (In fact, for every object e of EQ_Y we have constructed a morphism $k : e \longrightarrow e'$ in EQ_Y with e' having monomorphic solution m.)

Now we have defined RY as the union of all $(e')^\dagger = m : X' \longrightarrow TY$, and since this union is filtered, it is a colimit of the corresponding filtered diagram (whose morphisms are all existing subobject inclusions, or, equivalently, all morphisms of EQ_Y, see Remark 3.8(ii)); this is true in every LFP category, see Adámek and Rosický (1994, 1.63). That diagram is cofinal in Eq_Y, whence Eq_Y has the same colimit, that is, $RY = \operatorname{colim} \mathrm{Eq}_Y$. □

Notation 4.7.

(i) We use

$$e^\sharp : X \longrightarrow RY \quad (\text{for all } e : X \longrightarrow HX + Y \text{ in } \mathsf{EQ}_Y)$$

to denote the colimit cocone of Eq_Y.

(ii) Observe that for any finitely presentable object Y of $\mathscr{A}$, $\mathsf{inr} : Y \longrightarrow HX + Y$ is an object of EQ_Y. We use

$$\eta^R_Y : Y \longrightarrow RY$$

to denote the colimit morphism $\mathsf{inr}^\sharp$.

Example 4.8. Let $H = H_\Sigma$ be a polynomial endofunctor of Set. Then RY is the algebra of all rational trees over Y, and for every finitary flat equation system $e : X \longrightarrow H_\Sigma X + Y$ the colimit morphism $e^\sharp : X \longrightarrow RY$ is a codomain restriction of the solution $e^\dagger : X \longrightarrow TY$.

Remark 4.9. Notice that for every finitely presentable object Y of $\mathscr{A}$ the monomorphism

$$\varepsilon_Y : RY \longrightarrow TY$$

of Definition 4.2 makes the triangles

$$\begin{array}{ccc} X & & \\ {\scriptstyle e^\sharp}\downarrow & \searrow^{e^\dagger} & \\ RY & \xrightarrow[\varepsilon_Y]{} & TY \end{array}$$

commutative for all $e : X \longrightarrow HX + Y$ in EQ_Y. This is obvious if $e^\dagger$ is monomorphic; otherwise use the fact established in the proof of Proposition 4.6 that the finitary flat equation morphisms with monomorphic solutions form a cofinal subcategory of EQ_Y.

This makes ε_Y uniquely determined by the equations $\varepsilon_Y e^\sharp = e^\dagger$.

Proposition 4.10. **($RY = HRY + Y$.)** For every finitely presentable object Y of $\mathscr{A}$ the diagram Eq_Y has the colimit

$$\operatorname{colim} \mathrm{Eq}_Y = HRY + Y$$

with the following colimit cocone

$$X \xrightarrow{e} HX + Y \xrightarrow{He^\sharp + Y} HRY + Y \quad (\text{for } e : X \longrightarrow HX + Y \text{ in } \mathsf{EQ}_Y)$$

Proof.

(i) The morphisms $\delta_e = (He^\sharp + Y) \cdot e$ form a cocone of Eq_Y, that is, for every morphism

$$\begin{array}{ccc} X & \xrightarrow{e} & HX + Y \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh+Y} \\ X' & \xrightarrow[e']{} & HX' + Y \end{array}$$

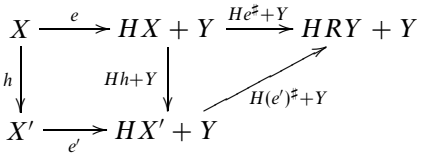
of EQ_Y we have

$$\delta_e = \delta_{e'} \cdot h.$$

Indeed, since h is a morphism, we know that

$$e^\sharp = (e')^\sharp \cdot h.$$

Therefore, the diagram



commutes.

(ii) In order to complete the proof, we shall show that the unique morphism

$$i : RY \longrightarrow HRY + Y \quad \text{with } i \cdot e^\sharp = \delta_e \text{ (for } e \text{ in } \mathsf{EQ}_Y)$$

is an isomorphism. Define $j : HRY + Y \longrightarrow RY$ as follows. Since H preserves filtered colimits, we have

$$HRY = \operatorname{colim} H\mathrm{Eq}_Y,$$

and therefore

$$HRY + Y = \operatorname{colim}(H\mathrm{Eq}_Y + Y)$$

with colimit cocone $He^\sharp + Y$ (e in EQ_Y). Define for every

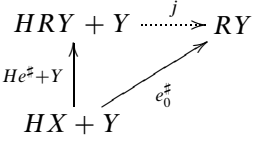
$$e : X \longrightarrow HX + Y \quad \text{in } \mathsf{EQ}_Y$$

a new member

$$e_0 \equiv\ HX + Y \xrightarrow{He+Y} H(HX + Y) + Y$$

of EQ_Y. Note that $HX + Y$ lies in $\mathscr{A}_{fp}$ since H is strongly finitary and $\mathscr{A}_{fp}$ is closed under coproducts.

Define a morphism $j : HRY + Y \longrightarrow RY$ by the commutativity of the following triangles:



for all e in EQ_Y.

The morphism j is well-defined since $e_0^\sharp$ (for e in EQ_Y) form a cocone. In fact, for every morphism $h : e \longrightarrow e'$ in EQ_Y, it is easy to show that $Hh + Y : e_0 \longrightarrow e_0'$ is also a morphism of EQ_Y. But then

$$e_0^\sharp = (e_0')^\sharp \cdot (Hh + Y),$$

as desired.

(ii a) Proof of $j \cdot i = id$: It is our task to show that

$$jie^\sharp = e^\sharp \quad \text{for all } e \text{ in } \mathsf{EQ}_Y.$$

For this observe that e is a morphism in EQ_Y from e to e_0. Therefore

$$j \cdot i \cdot e^\sharp = j \cdot (He^\sharp + Y) \cdot e = e_0^\sharp \cdot e = e^\sharp.$$

(ii b) Proof of $i \cdot j = id$: Here we must show that

$$i \cdot j \cdot (He^\sharp + Y) = He^\sharp + Y \quad \text{for all } e \text{ in } \mathsf{EQ}_Y.$$

But this is again easily seen:

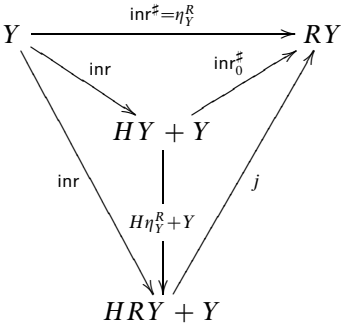
$$i \cdot j \cdot (He^\sharp + Y) = i \cdot e_0^\sharp = (He_0^\sharp + Y) \cdot (He + Y) = He^\sharp + Y$$

where the last step again uses $e : e \longrightarrow e_0$ in EQ_Y. □

Remark 4.11. We have established that

$$RY = HRY + Y \quad (Y \text{ in } \mathscr{A}_{fp})$$

with coproduct injection given by $j \cdot \mathsf{inl}$ and $j \cdot \mathsf{inr}$. Note that $j \cdot \mathsf{inl} = \eta_Y^R : Y \longrightarrow RY$. Indeed, the diagram

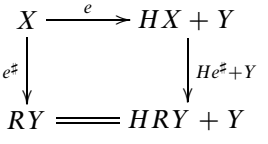


commutes.

Notation 4.12. From now on we shall use for any finitely presentable object Y the following notation for the coproduct injections:

$$\rho_Y : HRY \longrightarrow RY \quad \text{and} \quad \eta_Y^R : Y \longrightarrow RY.$$

Corollary 4.13. Let Y be a finitely presentable object. For every finitary flat equation morphism $e : X \longrightarrow HX + Y$, the diagram



commutes.

Proof. In fact, to show that

$$e^\sharp = \left[\rho_Y, \eta_Y^R\right] \cdot (He^\sharp + Y) \cdot e,$$

just use the fact that $[\rho_Y, \eta_Y^R] = j$ and that (since $e : e \longrightarrow e_0$ is a morphism in EQ_Y), $e^\sharp = e_0^\sharp \cdot e$. Thus,

$$e^\sharp = e_0^\sharp \cdot e = \left[\rho_Y, \eta_Y^R\right] \cdot (He^\sharp + Y) \cdot e$$

□

Remark 4.14.

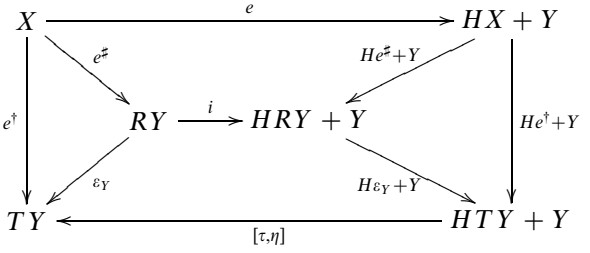
(i) For any finitely presentable object Y, we have

$$\varepsilon_Y = H\varepsilon_Y + Y.$$

More precisely, if $i : RY \longrightarrow HRY + Y$ denotes the isomorphism of Proposition 4.10, then its composite with $H\varepsilon_Y + Y : HRY + Y \longrightarrow HTY + Y = TY$ is equal to ε_Y. This follows from the equalities

$$\varepsilon_Y \cdot e^\sharp = (H\varepsilon_Y + Y) \cdot i \cdot e^\sharp \quad (e \text{ in } \mathsf{EQ}_Y).$$

In fact, consider the following diagrams:



The upper square commutes by definition of i, the two outer triangles by Remark 4.9. The outer square commutes since the solution $e^\dagger$ of e is given by corecursion, see Proposition 3.5. Thus, the lower square commutes, too.

(ii) Consequently, ε_Y is a homomorphism of H-algebras:

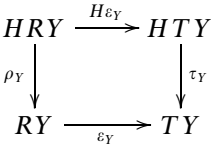
$$\varepsilon_Y \cdot \rho_Y = \tau_Y \cdot H\varepsilon_Y$$

with

$$\varepsilon_Y \cdot \eta_Y^R = \eta_Y. \tag{5}$$

In fact, $\varepsilon_Y \cdot \eta_Y^R = \mathsf{inr}^\dagger$ for $\mathsf{inr} : Y \longrightarrow HY + Y$, and it is easy to verify that $\mathsf{inr}^\dagger = \eta_Y$.

Corollary 4.15. For every finitely presentable object Y, the square



is a pullback.

Proof. In fact, since $\mathscr{A}$ is extensive, for every morphism $f : A \longrightarrow B$ the squares

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ {\scriptstyle \mathsf{inl}}\downarrow & & \downarrow{\scriptstyle \mathsf{inl}} \\ A + Y & \xrightarrow[f+Y]{} & B + Y \end{array}$$

are pullbacks. Apply this to $f = H\varepsilon_Y$. □

Theorem 4.16 (Rational Substitution Theorem). For every morphism

$$s : X \longrightarrow RY \quad (X,\ Y \text{ finitely presentable}),$$

there exists a unique extension to a homomorphism

$$\widetilde{s} : RX \longrightarrow RY$$

of H-algebras (that is, a unique homomorphism with $s = \widetilde{s} \cdot \eta^R_Y$).

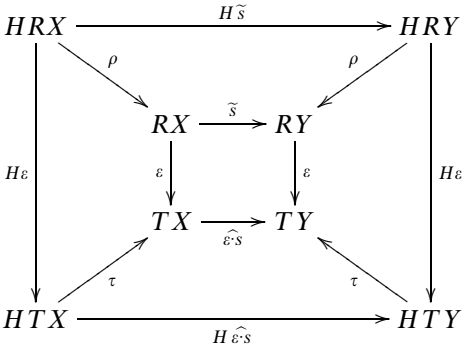
Proof. (I) **Existence:** We are going to find a morphism $\widetilde{s}$ such that the square

$$\begin{array}{ccc} RX & \xrightarrow{\widetilde{s}} & RY \\ {\scriptstyle \varepsilon_X}\downarrow & & \downarrow{\scriptstyle \varepsilon_Y} \\ TX & \xrightarrow[\widehat{\varepsilon_Y \cdot s}]{} & TY \end{array} \tag{6}$$

commutes. It follows from Remark 4.14(ii) and Theorem 1.7 that

$$\varepsilon_Y \cdot \widetilde{s} \cdot \eta^R_Y = \widehat{\varepsilon_Y \cdot s} \cdot \eta_X = \varepsilon_Y \cdot s,$$

which, since ε_Y is a monomorphism by definition, proves that $\widetilde{s}$ extends s. It also follows that $\widetilde{s}$ is a homomorphism of H-algebras: to conclude $\widetilde{s} \cdot \rho_X = \rho_Y \cdot H\,\widetilde{s}$, use the fact that ε_Y is a monomorphism and that the diagram

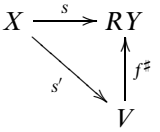


commutes (because the inner square is the above square (6), the outer one is its image and $\widehat{\varepsilon_Y \cdot s}$, ε_X and ε_Y are homomorphisms of H-algebras).

In order to define $\widetilde{s}$, use the fact that $RY = \operatorname{colim} \mathrm{Eq}_Y$ is a filtered colimit and X is finitely presentable. Thus, there exists an object

$$f : V \longrightarrow HV + Y \quad \text{of } \mathsf{EQ}_Y$$

such that s factors through its colimit morphism:



This allows us to define the desired morphism $\widetilde{s} : \operatorname{colim} \mathrm{Eq}_X \longrightarrow RY$ by providing a 'suitable' cocone of the diagram Eq_X as follows: Every object of EQ_X, say, $e : Z \longrightarrow HZ + X$, will be turned into an object $\overline{e}$ of EQ_Y by adding V (the domain of f) as new variables: $\overline{e} : Z + V \longrightarrow H(Z + V) + Y$. In more detail, for every object

$$e : Z \longrightarrow HZ + X \quad \text{of } \mathsf{EQ}_X$$

we use

$$\bar{e} : Z + V \longrightarrow H(Z + V) + Y$$

to denote the object of EQ_Y with the following components:

$$\bar{e} \cdot \mathsf{inl} \equiv Z \xrightarrow{e} HZ + X \xrightarrow{HZ + fs'} HZ + HV + Y \xrightarrow{[H\mathsf{inl}, H\mathsf{inr}] + Y} H(Z + V) + Y \quad (7)$$

$$\bar{e} \cdot \mathsf{inr} \equiv V \xrightarrow{f} HV + Y \xrightarrow{H\mathsf{inr} + Y} H(Z + V) + Y\,.$$

We shall show below that the morphisms

$$Z \xrightarrow{\mathsf{inl}} Z + V \xrightarrow{\bar{e}^\sharp} RY \quad (e \text{ in } \mathsf{EQ}_X) \tag{8}$$

form a cocone of Eq_X. This establishes the unique

$$\widetilde{s} : RX \longrightarrow RY \quad \text{with } \widetilde{s} \cdot e^\sharp = \bar{e}^\sharp \cdot \mathsf{inl} \quad (e \text{ in } \mathsf{EQ}_X) \tag{9}$$

This morphism $\widetilde{s}$ fulfills

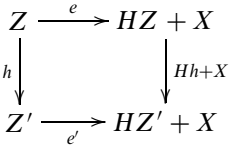
$$\varepsilon_Y \cdot \widetilde{s} \cdot e^\sharp = (\varepsilon_Y \cdot \bar{e}^\sharp) \cdot \mathsf{inl} = \bar{e}^\dagger \cdot \mathsf{inl}. \tag{10}$$

We finally establish that

$$\bar{e}^\dagger \cdot \mathsf{inl} = \widehat{\varepsilon_Y \cdot s} \cdot e^\dagger = \widehat{\varepsilon_Y \cdot s} \cdot \varepsilon_X \cdot e^\sharp, \tag{11}$$

which proves, together with (10), that (6) commutes, since both sides are equal when composed with the injections $e^\sharp$ of the colimit RX. This concludes the proof.

Proof that (8) is a cocone. Suppose that a morphism of EQ_X is given:



We are going to prove the commutativity of the following square:

$$\begin{array}{ccc} Z+V & \xrightarrow{\bar{e}} & H(Z+V)+Y \\ {\scriptstyle h+V}\downarrow & & \downarrow{\scriptstyle H(h+V)+Y} \\ Z'+V & \xrightarrow[\bar{e'}]{} & H(Z'+V)+Y \end{array} \tag{12}$$

That means that $h+V$ is a morphism from $\bar{e}$ to $\bar{e'}$ in EQ_Y, proving $\bar{e}^\sharp = \bar{e'}^\sharp \cdot (h+V)$, which establishes

$$\bar{e}^\sharp \cdot \mathsf{inl} = \bar{e'}^\sharp \cdot \mathsf{inl} \cdot h,$$

as desired.

We consider, in (12), the components of $Z+V$ separately. The equality of the right-hand components of (12) is obvious form the definition of $\bar{e}$ and $\bar{e'}$. For the left-hand components, consider the following squares:

$$\begin{array}{ccccccc} Z & \xrightarrow{e} & HZ+X & \xrightarrow{HZ+fs'} & HZ+HV+Y & \xrightarrow{[H\mathsf{inl},H\mathsf{inr}]+Y} & H(Z+V)+Y \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh+X} & & & & \downarrow{\scriptstyle H(h+V)+Y} \\ Z' & \xrightarrow[e']{} & HZ'+X & \xrightarrow[HZ'+fs']{} & HZ'+HV+Y & \xrightarrow[[H\mathsf{inl},H\mathsf{inr}]+Y]{} & H(Z'+V)+Y \end{array}$$

The left-hand square commutes by assumption and the right-hand one obviously does. Hence, the outer square commutes too.

Proof of (11). We shall now show that $\bar{e}^\dagger = [\widehat{\varepsilon_Y s} \cdot e^\dagger, f^\dagger]$, which establishes (11) above. By Proposition 3.5, we have to show that the square

$$\begin{array}{ccc} Z+V & \xrightarrow{\bar{e}} & H(Z+V)+Y \\ {\scriptstyle [\widehat{\varepsilon_Y s}\, e^\dagger, f^\dagger]}\downarrow & & \downarrow{\scriptstyle H[\widehat{\varepsilon_Y s}\, e^\dagger, f^\dagger]+Y} \\ TY & = & HTY+Y \end{array} \tag{13}$$

commutes.

We consider the components of the coproduct $Z+V$ separately. For the right-hand component of (13), apply Proposition 3.5 again to obtain

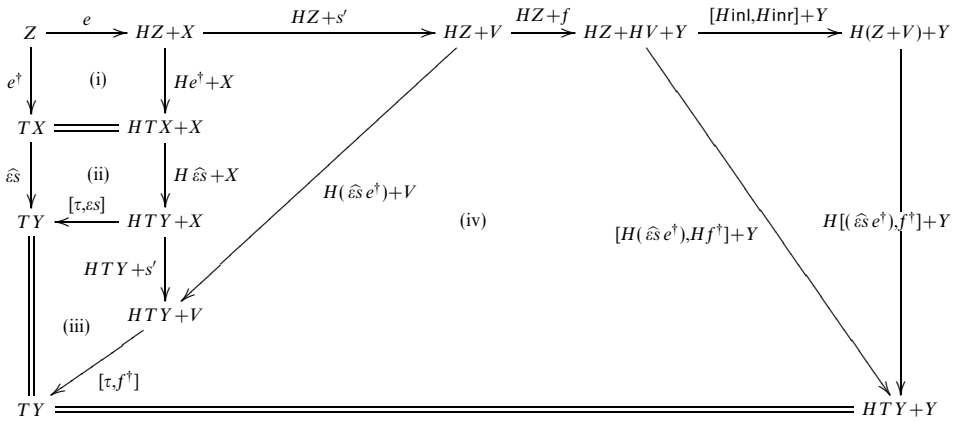
$$f^\dagger \equiv V \xrightarrow{f} HV+Y \xrightarrow{Hf^\dagger+Y} HTY+Y = TY$$

thus, the diagram

$$\begin{array}{ccccc} V & \xrightarrow{f} & HV+Y & \xrightarrow{H\mathsf{inr}+Y} & H(Z+V)+Y \\ {\scriptstyle f^\dagger}\downarrow & & & \searrow^{Hf^\dagger+Y} & \downarrow{\scriptstyle H[\widehat{s}\, e^\dagger, f^\dagger]+Y} \\ TY & = & = & = & HTY+Y \end{array}$$

commutes.

For the left-hand component of (13), consider the following commutative diagram:



Square (i) commutes once more by Proposition 3.5. For square (ii), use the definition of $\widehat{(\_)}$, see Substitution Theorem 1.7: we have

$$\widehat{\varepsilon_Y s}\,[\tau_X, \eta_X] = [\tau_Y H(\widehat{\varepsilon_Y s}), \varepsilon_Y s] = [\tau_Y, \varepsilon_Y s](H(\widehat{\varepsilon_Y s}) + X).$$

Square (iii) commutes since

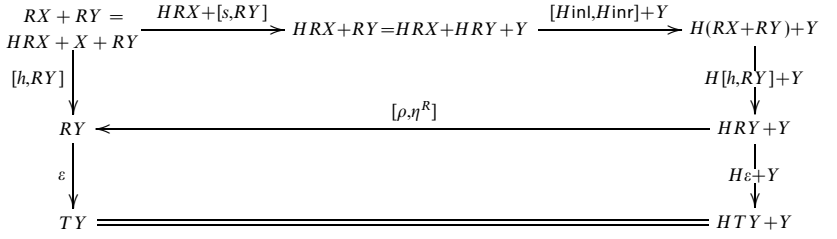
$$\varepsilon_Y \cdot s = \varepsilon_Y \cdot f^{\sharp} \cdot s' = f^{\dagger} \cdot s'. \tag{14}$$

The right-hand component of (iv) commutes by Proposition 3.5, and the left-hand one does trivially. The two remaining triangles obviously commute. Hence the outer square commutes as desired.

(II) **Uniqueness:** We are going to prove that for every homomorphism of H-algebras $h : RX \longrightarrow RY$ extending s, the following identity holds:

$$\varepsilon_Y [h, RY] = \varepsilon_Y [\widetilde{s}, RY]. \tag{15}$$

Since ε_Y is a monomorphism, it follows that $h = \widetilde{s}$. Observe that the diagram



(16)

commutes. In fact, the lower square commutes because $\tau_Y \cdot H\varepsilon_Y = \varepsilon_Y \cdot \rho_Y$ (ε_Y is a homomorphism with $\varepsilon_Y \cdot \eta_Y^R = \eta_Y$ by Remark 4.14(ii)). For the upper square consider the three components of $HRX + X + RY$: the left-hand component commutes because $\rho_Y \cdot Hh = h \cdot \rho_X$ (h is a homomorphism by assumption), the middle component does because h extends s, that is,

$$s = h \cdot \eta_X^R,$$

and the right-hand component commutes trivially. Thus, $\varepsilon_Y[h, RY]$ is a homomorphism into the final coalgebra TY. This holds for all H-algebra homomorphisms extending s, and, in particular, for $\widetilde{s}$. By coinduction, we conclude (15). □

Corollary 4.17. We obtain a finitary Kleisli triple $(R, \eta^R, \widetilde{(\_)})$ on $\mathscr{A}$.

Proof. In fact, the axioms (i)–(iii) of Definition 2.4 follow immediately from the uniqueness of $\widetilde{(\_)}$:

(i) $\widetilde{s} \cdot \eta_X = s$ by definition of $\widetilde{s}$.
(ii) $\widetilde{\eta_X} = id$ because id is a homomorphism extending η_X.
(iii) $\widetilde{r}\,\widetilde{s} = \widetilde{\widetilde{r}\,s}$ because $\widetilde{r}\,\widetilde{s}$ is a homomorphism extending $\widetilde{r}\,s$: from $\widetilde{s}\,\eta_X = s$ derive

$$(\widetilde{r}\,\widetilde{s})\eta_X = \widetilde{r}\,s$$

□

Definition 4.18. The finitary monad R defined by the above finitary Kleisli triple is called a *rational monad* generated by H.

Notation 4.19. The unit of the rational monad is denoted by

$$\eta^R : Id \longrightarrow R,$$

observe that for Y finitely presentable, η^R_Y are the above morphisms $\mathsf{inr}^\sharp$. We further use

$$\mu^R : RR \longrightarrow R$$

to denote the multiplication, given by

$$\mu^R_Y = \widetilde{id_{RY}} : RRY \longrightarrow RY$$

for finitely presentable Y.

The morphisms $\rho_Y : HRY \longrightarrow RY$ for Y finitely presentable in Notation 4.12 yield a natural transformation

$$\rho : HR \longrightarrow R.$$

In fact, $\mathscr{A}$ is a free cocompletion of $\mathscr{A}_{fp}$ under filtered colimits (Adámek and Rosický 1994), therefore the above natural transformation $\rho_Y : HRY \longrightarrow RY$ for $Y \in \mathscr{A}_{fp}$ extends uniquely to $\rho : HR \longrightarrow R$. We have

$$R = HR + Id$$

with the above injections ρ and η^R. Since coproduct injections are monomorphisms, the components of the last two natural transformations are monomorphisms. And since H preserves monomorphisms, the natural transformation

$$\rho^* \equiv H \xrightarrow{H\eta^R} HR \xrightarrow{\rho} R$$

also has monomorphic components. This is analogous to $\tau^* : H \longrightarrow T$ in Notation 3.2.

Remark 4.20. The fact that RY is a filtered union of all images of solutions of finitary flat equations extends from finitely presentable objects Y to all objects. In fact, both of the formulas

$$RY = \bigcup \mathrm{im}(e^\dagger)$$

and

$$RY = \operatorname{colim} \mathrm{Eq}_Y$$

extend from finitely presentable objects Y to all objects. Given any object Y in $\mathscr{A}$, we use, again,

$$\mathsf{EQ}_Y$$

to denote the category of all coalgebras of $H(\_)+Y$ carried by finitely presentable objects X and by

$$\mathrm{Eq}_Y : \mathsf{EQ}_Y \longrightarrow \mathscr{A}$$

the forgetful functor. This is a filtered diagram and we show that it has colimit

$$RY = \operatorname{colim} \mathrm{Eq}_Y .$$

Express $Y = \operatorname{colim}_{t\in I} Y_t$ as a filtered colimit of finitely presentable objects Y_t with colimit cocone $y_t : Y_t \xrightarrow{t\in I} Y$, $t \in I$. Then each $e : X \longrightarrow HX + Y = \operatorname{colim}_{t\in I}(HX + Y_t)$ factors as

$$X \xrightarrow{e_0} HX + Y_t \xrightarrow{HX+y_t} HX + Y$$

for some $t \in I$. We have $e_0^\sharp : X \longrightarrow RY_t$ as in Notation 4.7(i). Since R is finitary, we have $RY = \operatorname{colim}_{t\in I} RY_t$ with colimit injections $Ry_t : RY_t \longrightarrow RY$. We put

$$e^\sharp \equiv X \xrightarrow{e_0^\sharp} RY_t \xrightarrow{Ry_t} RY$$

This is independent of the above factorisation of e. To verify this, we use the fact that the diagram (Y_t) is filtered: it is sufficient to prove that for every connecting morphism $y_{t,t'} : Y_t \longrightarrow Y_{t'}$, if e_1 is defined by the commutative triangle

$$\begin{array}{ccc} X & \xrightarrow{e_0} & HX + Y_t \\ & \searrow^{e_1} & \downarrow{\scriptstyle HX+y_{t,t'}} \\ & & HX + Y_{t'} \end{array} \tag{17}$$

we have

$$e^\sharp \equiv X \xrightarrow{e_1^\sharp} RY_{t'} \xrightarrow{Ry_{t'}} RY .$$

In fact, since $Ry_{t'} = Ry_{t,t'} \cdot Ry_t$, it is sufficient to prove

$$e_1^\sharp = Ry_{t,t'} \cdot e_0^\sharp . \tag{18}$$

Recall that

$$Ry_{t,t'} = \widetilde{s}$$

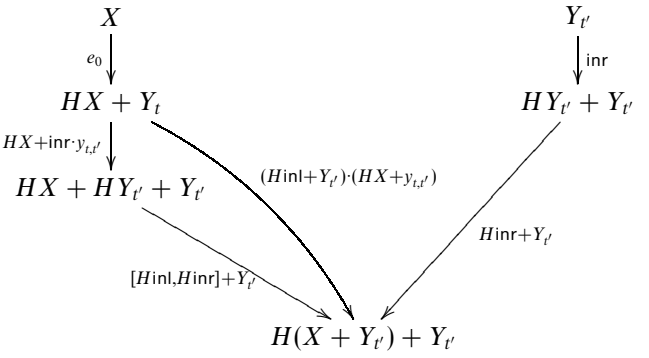
for

$$s \equiv Y_t \xrightarrow{y_{t,t'}} Y_{t'} \xrightarrow{\eta^R} RY_{t'} .$$

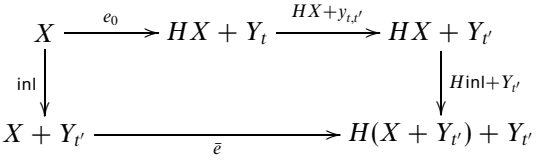
By (9),

$$Ry_{t,t'} \cdot e_0^\sharp = \bar{e}^\sharp \cdot \mathsf{inl} \tag{19}$$

where $\bar{e}$ has, by (7), the following components:



Consequently, the diagram

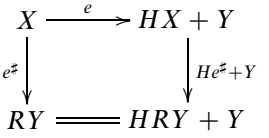


commutes. This proves that inl is a morphism of $\mathsf{EQ}_{Y_{t'}}$, thus, $\bar{e}^{\sharp} \cdot \mathsf{inl} = e_1^{\sharp}$, which together with (19) proves (18).

It is not difficult to see that the morphisms $e^{\sharp}$ form a cocone of Eq_Y. It follows from Proposition 4.6 that this is a colimit cocone. In fact, given a cocone $(c_e : X \longrightarrow C)_{e:X \longrightarrow HX+Y}$ of Eq_Y, we obtain a cocone of Eq_{Y_t} (for each t) by assigning to every $e_0 : X \longrightarrow HX + Y_t$ the morphism c_e, where $e = (HX + y_t)e_0$. This yields a unique $f_t : RY_t \longrightarrow C$, and we obtain a cocone of $(RY_t)_t$, which uniquely factors by way of $f : RY \longrightarrow C$. Then $fe^{\sharp} = c_e$ and f is unique with this property. Consequently, $RY = \operatorname{colim} \mathrm{Eq}_Y$.

The fact that this implies $RY = \bigcup \mathrm{im}(e^{\dagger})$ is proved exactly as in Proposition 4.6.

Lemma 4.21. For every finitary flat equation morphism $e : X \longrightarrow HX + Y$ the diagram



commutes.

Proof. See Corollary 4.13 for finitely presentable objects Y. The extension to arbitrary objects Y (by way of $e^{\sharp} = Ry_t \cdot e_0^{\sharp}$ above) is routine. □

Remark 4.22. R is a submonad of T (that is, a subobject in the category of all monads and homomorphisms).

Indeed, we use $c : T_0 \longrightarrow T$ to denote a finitary coreflection of T, see Corollary 2.8. From (5) and (6) above, the morphisms ε_Y form a homomorphisms of finitary

Kleisli-triples, or, equivalently, a homomorphism of finitary monads. Hence we obtain a monad homomorphism $\varepsilon = c \cdot \varepsilon_0 : R \longrightarrow T$. It is easy to check that for every finitary flat equation morphism $e : X \longrightarrow HX + Y$ (where Y is any object of $\mathscr{A}$), we have

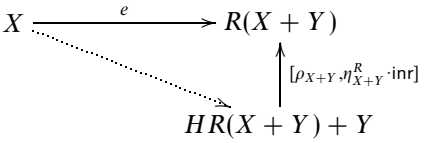
$$\varepsilon_Y e^{\sharp} = e^{\dagger}.$$

Just use the fact that $HX + Y$ is a filtered colimit of $HX + Y'$ for all $Y' \longrightarrow Y$ in $\mathscr{A}_{fp}/Y$, and since X is finitely presentable, e factors through some $e' : X \longrightarrow HX + Y'$, $Y' \in \mathscr{A}_{fp}$. Thus ε_Y is a monomorphism since Remark 4.20 shows that RY is the union of images of solutions of finitary flat equation morphisms.

We are going to prove that the rational monad is iterative (see Definition 1.16). The main part of this is the following solution theorem, dealing with *rational equation morphisms*, that is, morphisms

$$e : X \longrightarrow R(X + Y) \quad (X \text{ finitely presentable}),$$

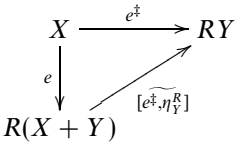
which are *guarded*, that is, factor through the coproduct injection of $R(X + Y) = HR(X + Y) + X + Y$:



Theorem 4.23 (Rational Solution Theorem). Every rational equation morphism $e : X \longrightarrow R(X + Y)$ has a unique solution. That is, there exists a unique morphism

$$e^{\ddagger} : X \longrightarrow RY$$

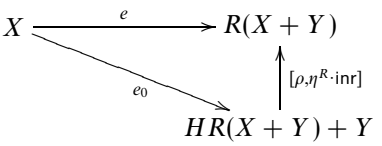
such that the triangle



(20)

commutes.

Proof. (I) **Existence:** Since e is guarded we have a factorisation



(21)

From the filtered colimit $RY = \operatorname{colim} \mathrm{Eq}_Y$ we obtain a filtered colimit

$$HR(X + Y) + Y = \operatorname{colim} H\mathrm{Eq}_{X+Y} + Y,$$

since $H(\_) + Y$ preserves filtered colimits. It follows that e_0, whose domain is finitely presentable, factors through some colimit arrow $Hg^\sharp + Y$, that is, there exists an object

$$g : W \longrightarrow HW + X + Y \quad \text{of } \mathsf{EQ}_{X+Y}$$

and a factorisation

$$\begin{array}{ccc} X & \xrightarrow{e_0} & HR(X+Y)+Y \\ & \searrow^{w} & \uparrow{\scriptstyle Hg^\sharp+Y} \\ & & HW+Y \end{array} \tag{22}$$

This defines an object

$$h \equiv W + X \xrightarrow{[g,\mathsf{inm}]} HW + X + Y \xrightarrow{[\mathsf{inl},w,\mathsf{inr}]} HW + Y \xrightarrow{H\mathsf{inl}+Y} H(W+X)+Y$$

of EQ_Y, since W and X are finitely presentable. (Here, inm denotes the injection into the middle summand.) We define

$$e^\ddagger \equiv X \xrightarrow{\mathsf{inr}} W + X \xrightarrow{h^\sharp} RY$$

and prove that (20) commutes. For that, consider the solution $h^\dagger : W + X \longrightarrow TY$ of h.

Note that since e is guarded for R, the equation morphism

$$\begin{array}{ccccc} X & \xrightarrow{e} & R(X+Y) & \xrightarrow{\varepsilon} & T(X+Y) \\ & \searrow^{e_0} & \uparrow{\scriptstyle [\rho,\eta^R\cdot\mathsf{inr}]} & & \uparrow{\scriptstyle [\tau,\eta\cdot\mathsf{inr}]} \\ & & HR(X+Y)+Y & \xrightarrow[H\varepsilon+Y]{} & HT(X+Y)+Y \end{array} \tag{23}$$

is guarded for T. In fact, the above square commutes, because ε_{X+Y} is a homomorphism with $\varepsilon_{X+Y} \cdot \eta^R_{X+Y} = \eta_{X+Y}$, see Remark 4.14(ii). We show that

$$(\varepsilon_{X+Y} \cdot e)^\dagger = h^\dagger \cdot \mathsf{inr}, \tag{24}$$

that is

$$(\varepsilon_{X+Y} \cdot e)^\dagger = \varepsilon_Y \cdot h^\sharp \cdot \mathsf{inr}.$$

This proves that the outer square of the diagram

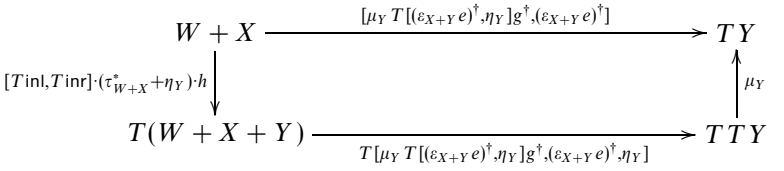
$$\begin{array}{ccccc} X & \xrightarrow{e^\ddagger} & RY & \xrightarrow{\varepsilon} & TY \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu^R} & & \uparrow{\scriptstyle \mu} \\ R(X+Y) & \xrightarrow{R[e^\ddagger,\eta^R]} & RRY & \xrightarrow{\varepsilon R} & \\ {\scriptstyle \varepsilon}\downarrow & \overset{T[e^\ddagger,\eta^R]}{\nearrow} & & \searrow^{T\varepsilon} & TRY \\ T(X+Y) & & \xrightarrow[T[\varepsilon e^\ddagger,\eta]]{} & & TTY \end{array} \tag{25}$$

commutes. Consequently, the upper left-hand square also commutes (which is the desired equality (20)) because the other inner parts of that diagram clearly commute (recall that ε is a monad morphism, see Remark 4.22), and ε is a monomorphism.

Proof of (24). We shall show that

$$h^\dagger = [\mu_Y T[(\varepsilon_{X+Y} e)^\dagger, \eta_Y] g^\dagger, (\varepsilon_{X+Y} e)^\dagger],$$

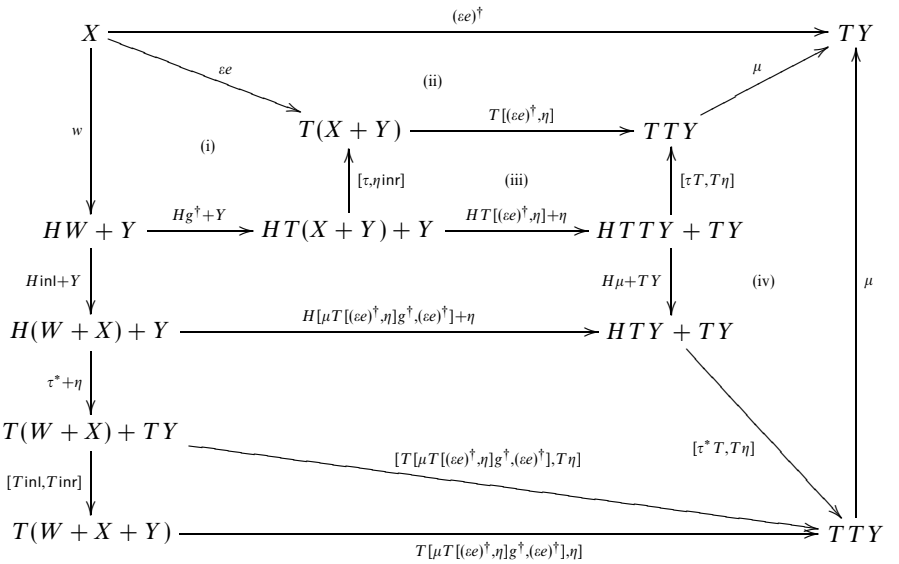
from which the required result follows. Thus it is our task to prove that the diagram



(26)

commutes.

We consider the components of the coproduct $W + X$ separately. For the right-hand one, we obtain the diagram (27):

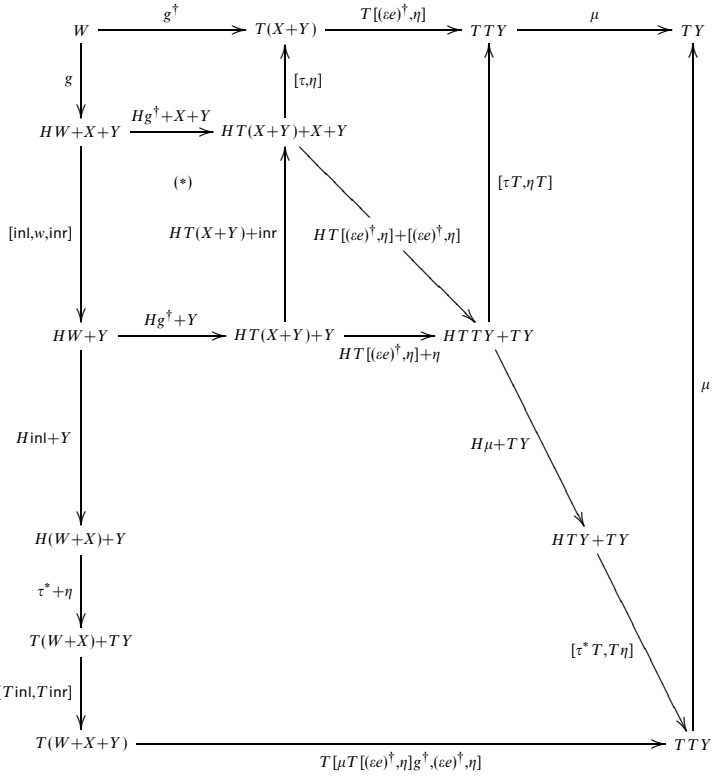


(27)

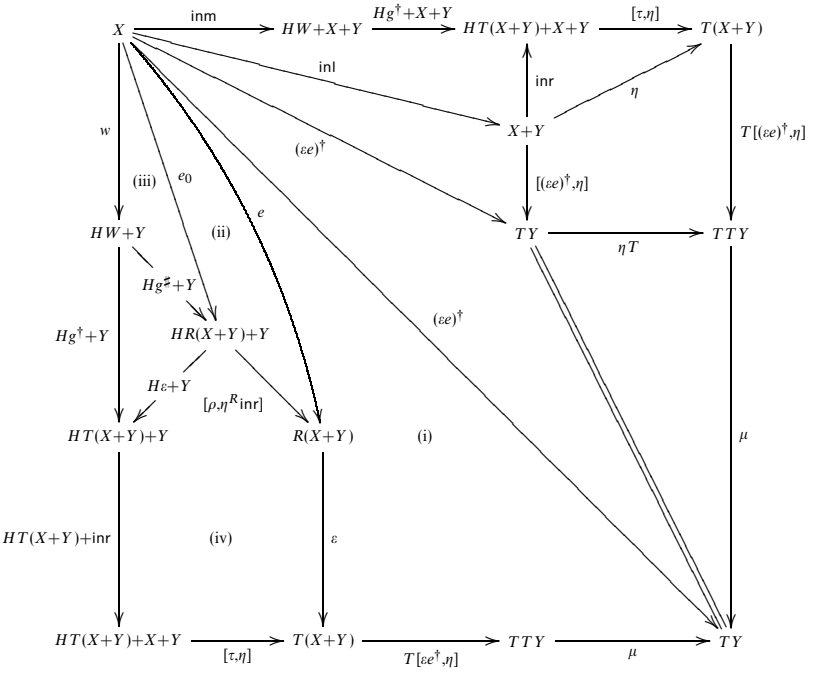
It commutes: in fact, square (i) commutes by (22) and (23), since $\varepsilon_{X+Y} g^\sharp = g^\dagger$. Square (ii) is just the definition of the solution $(\varepsilon_{X+Y} e)^\dagger$. Square (iii) is easily seen to commute when the components of $HT(X + Y) + Y$ are considered: for the left-hand components commutativity is obvious; for the right-hand ones we have

$$T[(\varepsilon_{X+Y} e)^\dagger, \eta_Y] \cdot \eta_{X+Y} \cdot \mathsf{inr} = T[(\varepsilon_{X+Y} e)^\dagger, \eta_Y] \cdot T\mathsf{inr} \cdot \eta_Y = T\eta_Y \cdot \eta_Y.$$

For (iv), recall that $\mu_Y \cdot \tau^*_{TY} = \tau_Y$, and then use the fact that μ_Y is an H-algebra homomorphism (see Corollary 1.8) in order to see that the left-hand components commute; for the right-hand ones commutativity is obvious. All the other parts of the above diagram clearly commute, and thus the whole diagram does.


W
g†
T(X+Y)
T[(εe)†,η]
TTY
μ
TY
g
[τ,η]
HW+X+Y
Hg†+X+Y
HT(X+Y)+X+Y
(*)
[τT,ηT]
[inl,w,inr]
HT(X+Y)+inr
HT[(εe)†,η]+[(εe)†,η]
HW+Y
Hg†+Y
HT(X+Y)+Y
HTTY+TY
HT[(εe)†,η]+η
μ
Hinl+Y
Hμ+TY
H(W+X)+Y
HTY+TY
τ*+η
T(W+X)+TY
[τ*T,Tη]
[Tinl,Tinr]
T(W+X+Y)
TTY
T[μT[(εe)†,η]g†,(εe)†,η]

(28)


X
inm
HW+X+Y
Hg†+X+Y
HT(X+Y)+X+Y
[τ,η]
T(X+Y)
inl
inr
η
X+Y
w
(εe)†
T[(εe)†,η]
(iii)
e0
e
[(εe)†,η]
HW+Y
(ii)
TY
ηT
TTY
Hg≅+Y
(εe)†
Hg†+Y
HR(X+Y)+Y
Hε+Y
[ρ,η^R inr]
HT(X+Y)+Y
R(X+Y)
(i)
μ
HT(X+Y)+inr
(iv)
ε
HT(X+Y)+X+Y
[τ,η]
T(X+Y)
T[εε†,η]
TTY
μ
TY

(29)

Finally, for the left-hand component of (26), we must show that the outward square of the diagram (28) commutes. Note that the lower part and the right-hand part of this diagram is the same as in the diagram (27), and therefore they commute. The upper left-hand square commutes by Proposition 3.5. Of the remaining three parts, we only need to consider square (∗); the other parts obviously commute. Consider the components of $HW + X + Y$ separately. The first and the last components obviously commute. The middle component need not commute *per se*, but it does when extended by the passage to TY in the upper right-hand corner of (28), which is sufficient for our purposes. In other words, we are to prove that the outward square of the diagram (29) commutes (here, inm denotes the injection into the middle summand).

In fact: (i) commutes by definition of $(\_)^\dagger$; (ii) commutes due to (21); (iii) does by (22); (iv) follows from ε_{X+Y} being a homomorphism of H-algebras (see Remark 4.14) and (5). The remaining inner parts commute trivially.

(II) **Uniqueness:** From the Solution Theorem 1.11, we know that $(\varepsilon_{X+Y} e)^\dagger$ is unique, and it is thus sufficient to verify that whenever $e^\ddagger$ is a solution in the sense of (20), we have

$$(\varepsilon_{X+Y} e)^\dagger = \varepsilon_Y e^\ddagger : X \longrightarrow TY.$$

Since ε_Y is a monomorphism by definition, this determines $e^\ddagger$. Thus, we just have to observe the commutativity of diagram (25). □

Remark 4.24. In the above proof we have seen that every rational equation morphism

$$e : X \longrightarrow R(X + Y)$$

reduces to a finitary flat equation morphism in the following sense: there is a finitely presentable object W (of 'additional' variables) and a finitary flat equation morphism

$$h : W + X \longrightarrow H(W + X) + Y$$

whose solution $h^\dagger : W + X \longrightarrow TY$ defines that of e via

$$e^\dagger = h^\dagger \mathsf{inr} : X \longrightarrow TY.$$

Or, equivalently, the rational solution $h^\ddagger : W + X \longrightarrow RY$ of h defines that of e via

$$e^\ddagger = h^\ddagger \mathsf{inr} : X \longrightarrow RY.$$

Corollary 4.25. The rational monad is iterative.

Proof. In fact, R is ideal due to Notation 4.19: we have

$$R = HR + Id$$

with injections ρ and η^R, and for the natural transformation

$$\mu^R : RR \longrightarrow R$$

with components $\mu^R_X = \widetilde{id_{RX}}$, we consider

$$H\mu^R : HRR \longrightarrow HR,$$

and find that the appropriate diagram

$$\begin{array}{ccc} HRR & \xrightarrow{H\mu^R} & HR \\ {\scriptstyle \rho R}\downarrow & & \downarrow{\scriptstyle \rho} \\ RR & \xrightarrow[\mu^R]{} & R \end{array}$$

commutes, since μ_X^R (being of the form $\widetilde{(\_)}$) is a homomorphism of H-algebras for every finitely presentable X.

Now the Rational Solution Theorem tells us that R is iterative, see Definition 1.16. □

Remark 4.26. Solutions for the monads R and T are closely related:

(i) For every guarded equation morphism $e : X \longrightarrow R(X + Y)$ for R, we have seen in the proof of Theorem 4.23 that $\varepsilon_{X+Y} \cdot e : X \longrightarrow T(X + Y)$ is also guarded, and the rational solution $e^\ddagger$ of e is determined by

$$\varepsilon_Y \cdot e^\ddagger = (\varepsilon_{X+Y} \cdot e)^\dagger.$$

(ii) Conversely, guarded equation morphisms $e : X \longrightarrow T(X + Y)$, which factor through $R(X + Y)$,

$$\begin{array}{ccc} X & \xrightarrow{e} & T(X+Y) \\ & {\scriptstyle e_0}\searrow & \uparrow{\scriptstyle \varepsilon} \\ & & R(X+Y) \end{array}$$

have the property that e_0 is guarded for R, and, therefore, the solution of e is determined by the rational solution of e_0:

$$e^\dagger = \varepsilon_Y \cdot e_0{}^\ddagger.$$

In fact, since e is guarded,

$$\begin{array}{ccc} X & \xrightarrow{e} & T(X+Y) \\ & {\scriptstyle e'}\searrow & \uparrow{\scriptstyle [\tau,\eta\cdot \mathsf{inr}]} \\ & & HT(X+Y)+Y \end{array}$$

we just have to observe that the square

$$\begin{array}{ccc} HR(X+Y)+Y & \xrightarrow{H\varepsilon+Y} & HT(X+Y)+Y \\ {\scriptstyle [\rho,\eta^R\cdot \mathsf{inr}]}\downarrow & & \downarrow{\scriptstyle [\tau,\eta\cdot \mathsf{inr}]} \\ R(X+Y) & \xrightarrow[\varepsilon]{} & T(X+Y) \end{array}$$

is a pullback. To verify this, observe that both the components of $HR(X+Y)+Y$ yield pullbacks: the left-hand one is the pullback in Corollary 4.15, for the right-hand one, use the fact that $\mathscr{A}$ is extensive, see Section 2.3. The universal property of pullbacks

yields, because of

$$[\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}] \cdot e' = e = \varepsilon_{X+Y} \cdot e_0,$$

the existence of $f : X \longrightarrow HR(X+Y)+Y$ with $e' = (H\varepsilon_{X+Y} + Y) \cdot f$ and

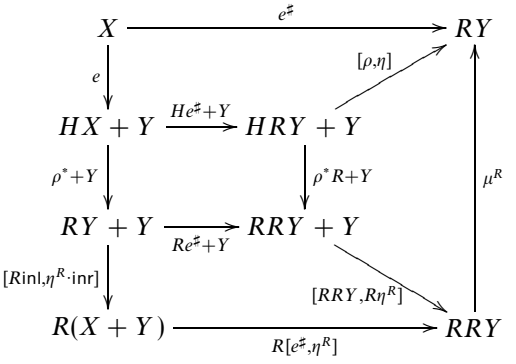
$$e_0 = [\rho, \eta^R \cdot \mathsf{inr}] \cdot f$$

which proves that e_0 is guarded with respect to R.

Remark 4.27. Every finitary flat equation morphism $e : X \longrightarrow HX + Y$ can be identified with the corresponding guarded rational equation morphism

$$X \xrightarrow{e} HX + Y \xrightarrow{\rho^* + Y} RX + Y \xrightarrow{[R\mathsf{inl}, \eta^R \cdot \mathsf{inr}]} R(X+Y).$$

The solution of the latter (in the sense of Theorem 4.23) is simply $e^\sharp$. In fact, the diagram



commutes; see Lemma 4.21 for the upper square.

Example 4.28. Let Σ be a signature with Σ_n finite, for every n, and $\Sigma_n = \emptyset$ for all $n \geqslant n_0$. Let $H = H_\Sigma$ be a (strongly finitary) polynomial functor in Set corresponding to Σ, see Section 1.2. Then TY is the Σ-algebra of finite and infinite Σ-labelled trees. We have constructed RY above by considering all images of solutions of finitary flat equations $e : X \longrightarrow H_\Sigma X + Y$ – that is, RY is the subalgebra of all rational trees in TY.

Thus, in the case of strongly finitary polynomial functors, the rational monad is the monad of rational trees. This was proved in Elgot *et al.* (1978) to be a free iterative monad on H.

Recall the notions of ideal and iterative monad from Definition 1.13 and Definition 1.16, respectively. We show that the rational monad R can be characterised as a free iterative monad, when ideal monad morphisms are taken as the 'right' morphisms.

Definition 4.29. Let S be an ideal monad.

(a) A natural transformation from a functor H to S is called *ideal* if it factors through $\sigma^S : S' \longrightarrow S$. For example, for the rational monad, the above natural transformation

$$\rho^* \equiv H \xrightarrow{H\eta^R} HR \xrightarrow{\rho} R$$

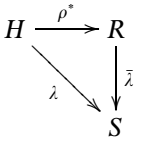
is ideal.

(b) Given another ideal monad $\bar{S}$, a monad morphism $\varphi : S \longrightarrow \bar{S}$ is called *ideal* if it has the form

$$\varphi = \varphi' + id$$

for some natural transformation $\varphi' : S' \longrightarrow \bar{S}'$.

Theorem 4.30. The rational monad R is a free iterative monad on H. That is, given an iterative monad S and an ideal transformation $\lambda : H \longrightarrow S$, there exists a unique ideal monad morphism $\bar{\lambda} : R \longrightarrow S$ for which the triangle



commutes.

Proof. (I) **Existence of** $\bar{\lambda}$: By Remark 2.9, it is sufficient to give the components

$$\bar{\lambda}'_Y : HRY \longrightarrow S'Y$$

of $\bar{\lambda}' : HR \longrightarrow S'$ (such that $\bar{\lambda} = \bar{\lambda}' + id$) for finitely presentable objects Y and to prove that for every arrow $s : Y \longrightarrow RZ$ (Y, Z finitely presentable) the square

$$\begin{array}{ccc} RY & \xrightarrow{\bar{s}} & RZ \\ {\scriptstyle \bar{\lambda}=\bar{\lambda}'+Y}\downarrow & & \downarrow{\scriptstyle \bar{\lambda}=\bar{\lambda}'+Z} \\ SY & \xrightarrow[\widehat{\bar{\lambda}s}]{} & SZ \end{array} \tag{30}$$

commutes (where $\widehat{\bar{\lambda}_Z s} = \mu^S_Z \cdot S(\bar{\lambda}_Z s)$, as usual).

To define $\bar{\lambda}'_Y$, whose domain is

$$HRY = \operatorname{colim} H\mathrm{Eq}_Y,$$

consider an arbitrary object $e : X \longrightarrow HX + Y$ of EQ_Y. Then the following, obviously guarded, equation morphism

$$\lambda \cdot He \equiv HX \xrightarrow{He} H(HX + Y) \xrightarrow{\lambda'} S'(HX + Y) \xrightarrow{\sigma} S(HX + Y) \tag{31}$$

has the unique solution $(\lambda_Y \cdot He)^\dagger : HX \longrightarrow SY$. Since we have, by Remark 3.6, a factorisation through σ_Y, say $(\lambda \cdot He)^\dagger = \sigma_Y \cdot \check{e}$, we can define $\bar{\lambda}'_Y$ by

$$\bar{\lambda}'_Y \cdot He^\sharp = \check{e} \quad (\text{for all } e : X \longrightarrow HX + Y \text{ in } \mathsf{EQ}_Y). \tag{32}$$

This is well-defined because all $\check{e}$ form a cocone of $H\mathrm{Eq}_Y$. In fact, for every morphism $h : e \longrightarrow e'$ in EQ_Y, we have a commutative diagram

$$\begin{array}{ccccc} HX & \xrightarrow{He} & H(HX+Y) & \xrightarrow{\lambda} & S(HX+Y) \\ {\scriptstyle Hh}\downarrow & & {\scriptstyle H(Hh+Y)}\downarrow & & \downarrow{\scriptstyle S(Hh+Y)} \\ HX' & \xrightarrow[He']{} & H(HX'+Y) & \xrightarrow[\lambda]{} & S(HX'+Y) \end{array}$$

This implies $(\lambda \cdot e)^\dagger = (\lambda \cdot e')^\dagger \cdot Hh$, see Lemma 3.7. Since σ_Y is a monomorphism, we conclude

$$\check{e} = \check{e}' \cdot Hh,$$

as required.

In order to prove that (30) commutes, we are to show that for every object $e : X \longrightarrow HX + Y$ of EQ_Y, we have

$$\widehat{\bar{\lambda}_Z s}\ \bar{\lambda}_Y e^\sharp = \bar{\lambda}_Z\ \widetilde{s}\ e^\sharp.$$

As in the proof of Theorem 4.16, factor

$$s = f^\sharp s' \tag{33}$$

and define $\bar{e} : Z + V \longrightarrow H(Z+V) + Y$ by (7). That is, $\bar{e}$ has the following components:

$$\bar{e} \cdot \mathsf{inl} \equiv X \xrightarrow{e} HX + Y \xrightarrow{HX + fs'} HX + HV + Z \xrightarrow{[H\mathsf{inl}, H\mathsf{inr}]+Z} H(X+V) + Z$$

and

$$\bar{e} \cdot \mathsf{inr} \equiv V \xrightarrow{f} HV + Z \xrightarrow{H\mathsf{inr}+Z} H(X+V) + Z.$$

Note that for all e in EQ_Y, we have

$$\bar{\lambda}_Y e^\sharp = \left[(\lambda He)^\dagger, \eta_Y^S\right] \cdot e. \tag{34}$$

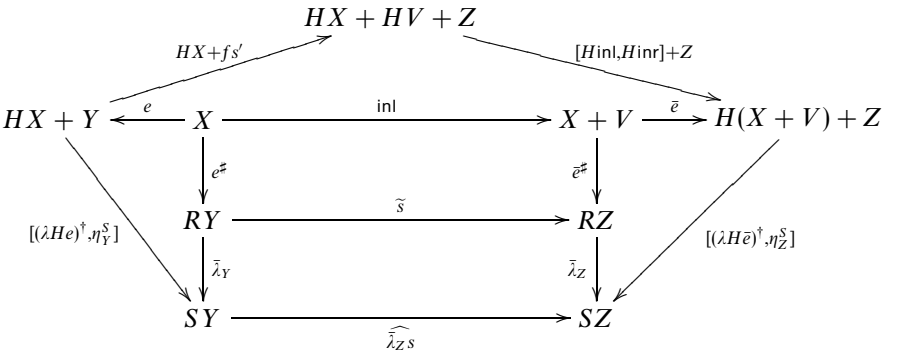
Indeed, consider the following diagram:

$$\begin{array}{ccccc} X & \xrightarrow{e} & HX+Y & = & HX+Y \\ {\scriptstyle e^\sharp}\downarrow & & \downarrow{\scriptstyle He^\sharp+Y} & & \\ RY & \xrightarrow[i]{\cong} & HRY+Y & & \downarrow{\scriptstyle [(\lambda He)^\dagger, \eta_Y^S]} \\ & {\scriptstyle \bar{\lambda}_Y}\searrow & \downarrow{\scriptstyle [\sigma_Y \bar{\lambda}'_Y, \eta_Y^S]} & & \\ & & SY & = & SY \end{array}$$

The upper square commutes by the definition of the isomorphism $i : RY \longrightarrow HRY + Y$ (see Remark 4.11), the lower triangle is the definition of $\bar{\lambda}_Y$ on the components of the coproduct $RY = HRY + Y$, and the right-hand square commutes, since

$$\sigma_Y \cdot \bar{\lambda}'_Y \cdot He^\sharp = (\lambda He)^\dagger.$$

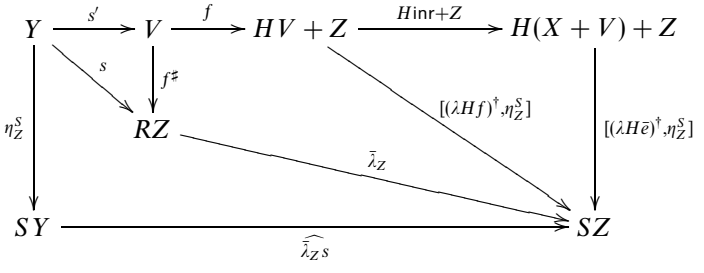
Now consider the following diagram:



(35)

Observe that the upper part commutes by the definition of $\bar{e}$, and the right-hand and left-hand parts commute by (34) for e and $\bar{e}$, respectively. Of the remaining two inner squares, the upper one commutes by definition of $\tilde{s}$, see (9). We shall show below that the outer shape commutes. Thus, the lower square commutes when precomposed with $e^\sharp$, for any e in EQ_Y, which establishes the commutativity of (30).

In order to show that the outer shape of (35) commutes, we consider the components of $HX + Y$ separately. For the right-hand one we obtain the following diagram:



(36)

Observe that all parts of this diagram, except perhaps the right-hand triangle, clearly commute: we have $\bar{\lambda}_Z f^\sharp = [(\lambda Hf)^\dagger, \eta^S_Z] \cdot f$ by (34) and the other two parts are obvious. Now notice that $\mathsf{inr} : f \longrightarrow \bar{e}$ is a morphism in EQ_Z. Therefore the square

$$\begin{array}{ccccc} HV & \xrightarrow{Hf} & H(HV+Z) & \xrightarrow{\lambda} & S(HV+Z) \\ {\scriptstyle H\mathsf{inr}}\downarrow & & & & \downarrow{\scriptstyle S(H\mathsf{inr}+Z)} \\ H(X+V) & \xrightarrow[H\bar{e}]{} & H(H(X+V)+Z) & \xrightarrow[\lambda]{} & S(H(X+V)+Z) \end{array}$$

commutes. Hence

$$(\lambda H\bar{e})^\dagger = (\lambda Hf)^\dagger \cdot H\mathsf{inr}$$

by Lemma 3.7, which establishes the commutativity of (36).

For the left-hand component of the outer shape of (35) we are to establish the commutativity of the following diagram:

$$\begin{array}{ccc} HX & \xrightarrow{H\mathsf{inl}} & H(X+V) \\ {\scriptstyle (\lambda He)^\dagger}\downarrow & & \downarrow{\scriptstyle (\lambda H\bar{e})^\dagger} \\ SY & \xrightarrow[\widehat{\bar{\lambda}_Z s}]{} & SZ \end{array} \quad \text{for all } e \text{ in } \mathsf{EQ}_Y \tag{37}$$

In order to prove this, consider the following arrow $h : HX + HV \longrightarrow H(HX + HV + Z)$ defined by

$$h \cdot \mathsf{inl} \equiv HX \xrightarrow{He} H(HX+Y) \xrightarrow{H(HX+fs')} H(HX+HV+Z)$$

and

$$h \cdot \mathsf{inr} \equiv HV \xrightarrow{Hf} H(HV+Z) \xrightarrow{H\mathsf{inr}} H(HX+HV+Z).$$

Note that we have

$$H([H\mathsf{inl}, H\mathsf{inr}] + Z) \cdot h = H\bar{e} \cdot [H\mathsf{inl}, H\mathsf{inr}]. \tag{38}$$

To show the commutativity of (37), we shall prove below that for the solution of the guarded equation morphism $\lambda_{HX+HV+Z} \cdot h$, the following two claims hold:

$$(\lambda h)^\dagger = (\lambda H\bar{e})^\dagger \cdot [H\mathsf{inl}, H\mathsf{inr}] \tag{39}$$

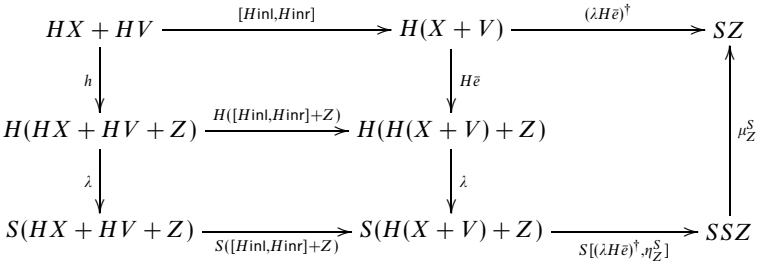
and

$$(\lambda h)^\dagger = [\widehat{\bar{\lambda}_Z s} \cdot (\lambda He)^\dagger, (\lambda Hf)^\dagger]. \tag{40}$$

Observe that the left-hand components yield (37), which concludes the proof of the existence of the ideal monad morphism $\bar{\lambda} : R \longrightarrow S$.

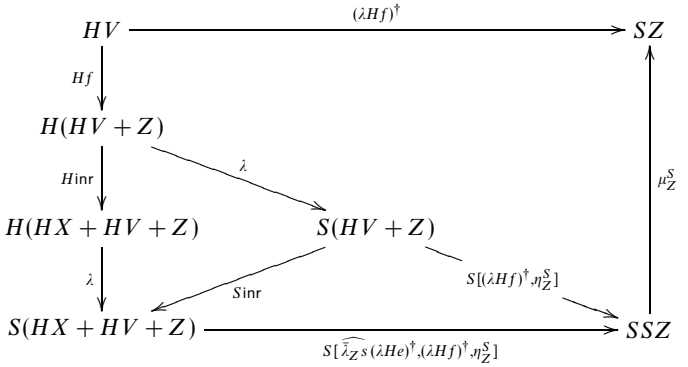
Proof of (39). Consider the commutative diagram (41). The upper left-hand square is just (38), the lower square commutes by naturality of λ. The right-hand square is the definition of $(\lambda H\bar{e})^\dagger$. Thus $(\lambda H\bar{e})^\dagger \cdot [H\mathsf{inl}, H\mathsf{inr}]$ solves λh, and thus (39) holds.

Proof of (40). As for (39), we use the definition of the solution $(\lambda h)^\dagger$, but we consider the



(41)

components of the coproduct $HX+HV$ separately. For the right-hand component of (40), consider the commutative diagram (42):



(42)

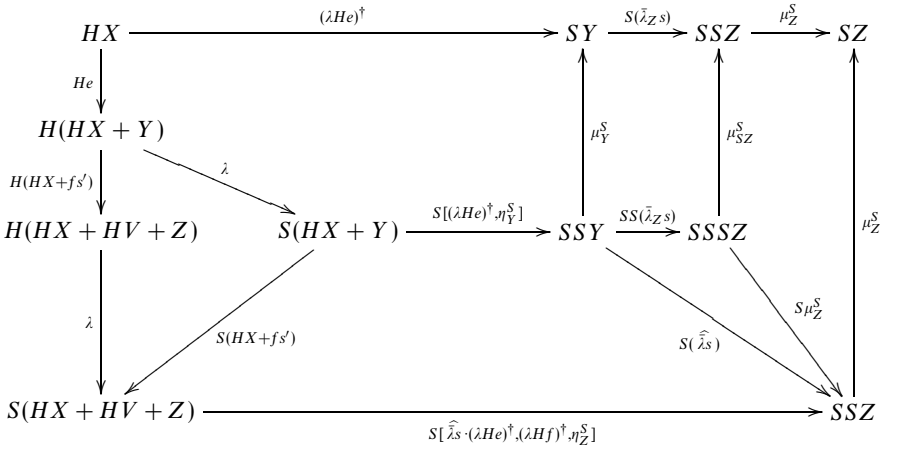
For the left-hand component of (40), we analyse the diagram (43). All inner parts of this diagram except, perhaps, the lower one obviously commute. For the lower part, remove S and consider the components of $HX + Y$. For the left-hand one, nothing needs to be shown, the right-hand one commutes since by (33) and (34) we have

$$\widehat{\bar{\lambda}_Z s} \cdot \eta^S_Y = \bar{\lambda}_Z \cdot s = \bar{\lambda}_Z \cdot f^\sharp \cdot s' = \left[(\lambda Hf)^\dagger, \eta^S_Z\right] \cdot f \cdot s'.$$

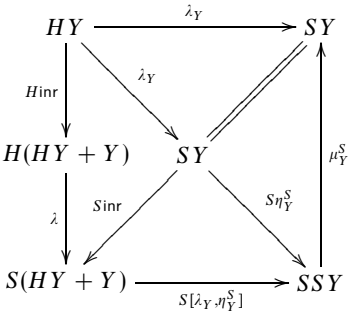
Therefore (43) commutes, concluding the proof of (40).

(II) **Equality** $\bar{\lambda} \cdot \rho^* = \lambda$**.** It suffices to check that $\bar{\lambda}_Y \cdot \rho^*_Y = \lambda_Y$ for objects Y in $\mathscr{A}_{fp}$, see Remark 2.2(iii). Notice that for $e = \mathsf{inr} : Y \longrightarrow HY + Y$ we have $(\lambda_{HY+Y} H\mathsf{inr})^\dagger = \lambda_Y$. Indeed, the diagram (44) commutes. That means we can choose $\lambda'_Y : HY \longrightarrow S'Y$ as our $\check{e}$ (see (32)). But now we have

$$\bar{\lambda}'_Y H\eta^R_Y = \bar{\lambda}'_Y H\mathsf{inr}^\sharp = \lambda'_Y.$$

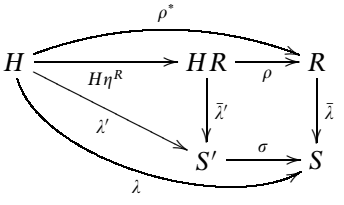


(43)



(44)

Extending by σ_Y, we obtain the desired equation. In fact, the diagram



commutes.

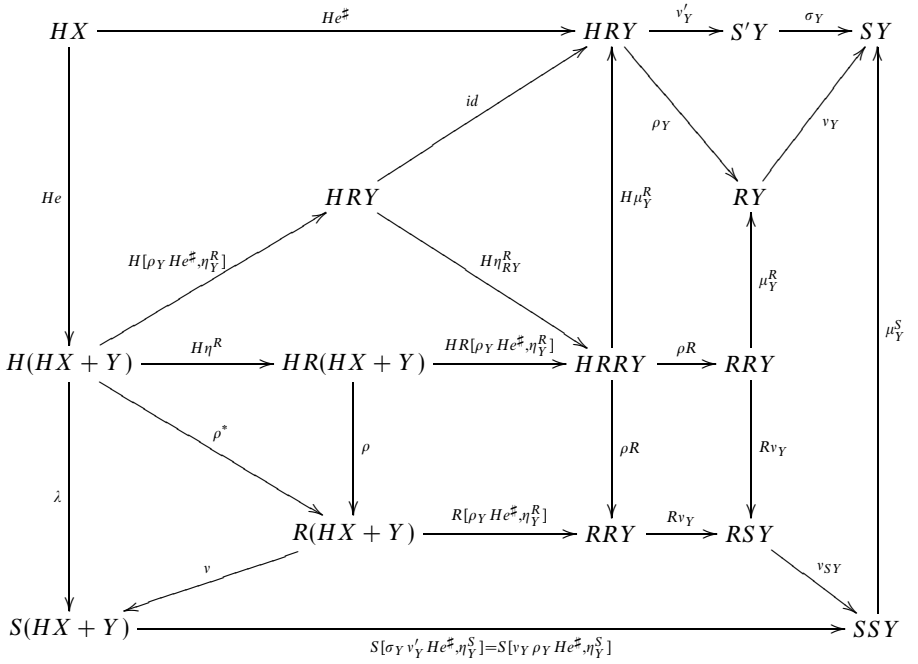
(III) **Uniqueness of $\bar{\lambda}$.** Suppose that $v = v' + id$ is another ideal monad morphism with $v \cdot \rho^* = \lambda$. We have to show that v'_Y satisfies the defining equation of $\bar{\lambda}'_Y$, that is,

$$v'_Y \cdot He^{\sharp} = \check{e} : HX \longrightarrow S'Y$$

for all objects Y of $\mathscr{A}_{fp}$ and e of EQ_Y. Since the coproduct injection $\sigma_Y : S'Y \longrightarrow SY$ is monomorphic, it suffices to show that

$$\sigma_Y \cdot v'_Y \cdot He^{\sharp} = \sigma_Y \cdot \check{e} = (\lambda He)^{\dagger}. \tag{45}$$

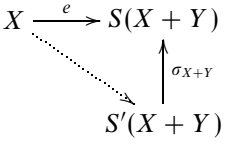
We use the definition of the solution $(\lambda He)^\dagger$ to show that this is indeed the case. All parts of the diagram (46) commute:



(46)

For the upper left-hand triangle, recall from Corollary 4.13 that $[\rho_Y, \eta^R_Y]\cdot(He^\sharp + Y)\cdot e = e^\sharp$, for the lower part use naturality of v and $v_Y \eta^R_Y = \eta^S_Y$, for the rest commutativity is obvious. This completes the proof of (45). □

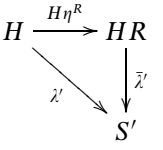
Remark 4.31. Recall that Elgot defined *iterative theories* by the (seemingly weaker) condition that all *ideal equation morphisms*, that is, all $e : X \longrightarrow S(X+Y)$ that factor through σ_{X+Y},



have unique solutions. This, however, does not influence the above result. That is, the rational monad is also a free iterative monad in the sense of Elgot. This follows from the above proof: the only guarded equation we used for the iterative monad S was the equation morphism $\lambda \cdot He$, see (31), and that is, indeed, ideal.

Remark 4.32. Our definition of ideal monad (as a monad S of the form $S = S' + Id$) rests on our assumption that coproduct injections are monomorphic in $\mathscr{A}$. Without this assumption, we would have to define an ideal monad by means of a functor $S' : \mathscr{A} \longrightarrow \mathscr{A}$ and natural transformations $\mu' : S'S \longrightarrow S'$ such that the functor $S = S' + Id$ together with the natural transformations $\mathsf{inr} : Id \longrightarrow S$ and $\mu \equiv SS = S'S + S \xrightarrow{[\mu', S]} S' + Id = S$ is

a monad. In that sense it is important to note that in the above theorem we actually proved the following:

> For every natural transformation $\lambda' : H \longrightarrow S'$, there exists a unique natural transformation $\bar{\lambda}' : HR \longrightarrow S'$ such that $\bar{\lambda}' + id$ is an ideal monad homomorphism and the triangle
>
> 
>
>
> commutes.

In fact, both the uniqueness, and the naturality of $\bar{\lambda}'_X : HRX \longrightarrow S'X$ follow from the uniqueness and naturality of $\bar{\lambda}_X : RX \longrightarrow SX$ (again through our assumptions that the coproduct injections ρ_X and σ_X are monomorphisms).

5. Rational monad – finitary case

Assumptions 5.1. Throughout this section $\mathscr{A}$ denotes a strongly LFP category and H a finitary endofunctor preserving monomorphisms.

We are going to prove that, once again, the rule

$$RY = \bigcup \mathrm{im}(e^\dagger)$$

defines a free iterative monad on H, called the rational monad of H.

Remark 5.2. We know from Corollary 2.16 that H is a directed colimit of strongly finitary functors preserving monomorphisms:

$$H = \operatorname*{colim}_{i \in I} H_i \quad (I \text{ a directed poset}).$$

We use

$$\alpha_{i,j} : H_i \longrightarrow H_j \quad (i \leqslant j)$$

to denote the connecting morphisms and

$$\beta_i : H_i \longrightarrow H \quad (i \in I)$$

to denote the colimit cocone.

Let

$$\rho_i^* : H_i \longrightarrow R_i \quad (i \in I)$$

be the rational monad of H_i as constructed in Section 4. For each $i \leqslant j$ we have an ideal natural transformation

$$H_i \xrightarrow{\alpha_{i,j}} H_j \xrightarrow{\rho_j^*} R_j$$

for which there exists, by Theorem 4.30, a unique extension to an ideal monad homomorphism $\bar{\alpha}_{i,j} : R_i \longrightarrow R_j$:

$$\begin{array}{ccc} H_i & \xrightarrow{\alpha_{i,j}} & H_j \\ {\scriptstyle \rho_i^*}\downarrow & & \downarrow{\scriptstyle \rho_j^*} \\ R_i & \xrightarrow[\bar{\alpha}_{i,j}]{} & R_j \end{array}$$

The unicity makes it clear that we obtain a directed diagram of iterative monads $(R_i \mid i \in I)$ (with connecting morphisms $\bar{\alpha}_{i,j}$), and we put

$$R = \operatorname*{colim}_{i \in I} R_i.$$

That is, we define an endofunctor R of $\mathscr{A}$ as a colimit (object-wise) of the endofunctors R_i, that is, $RY = \operatorname*{colim}_{i \in I} R_i Y$ for all Y in $\mathscr{A}$. Next we prove that a colimit of a filtered diagram of iterative monads is an iterative monad (Proposition 5.5). An easy consequence is that the above monad R is actually a free iterative monad on H. Finally, we prove that $RY = \operatorname{colim} \mathrm{Eq}_Y$.

Proposition 5.3 (Aczel *et al.* 2002). Ideal monad homomorphisms $\lambda : S \longrightarrow \bar{S}$ between iterative monads preserve solutions. That is, if $e : X \longrightarrow S(X + Y)$ is guarded for S and has the solution $e^\dagger : X \longrightarrow SY$, then $\lambda_{X+Y} \cdot e$ is guarded for $\bar{S}$ and has the solution $\lambda_Y \cdot e^\dagger$ with respect to $\bar{S}$.

Corollary 5.4. For all $i \leqslant j$ in I and any finitary flat equation $f : X \longrightarrow H_i X + Y$, put

$$g \equiv X \xrightarrow{f} H_i X + Y \xrightarrow{\alpha_{i,j}+Y} H_j X + Y.$$

Then the solution of f with respect to R_i denoted by $f_i^\sharp : X \longrightarrow R_i Y$ (see Remark 4.27), is related to the solution g_j of g with respect to R_j by

$$\begin{array}{ccc} & X & \\ {\scriptstyle f_i^\sharp}\swarrow & & \searrow{\scriptstyle g_j^\sharp} \\ R_i Y & \xrightarrow[\bar{\alpha}_{i,j}]{} & R_j Y \end{array}$$

Proof. Apply Proposition 5.3 to $\lambda = \bar{\alpha}_{i,j}$. □

Proposition 5.5. For every directed collection of iterative monads S_i ($i \in I$, I a directed poset) and ideal monad morphisms $\lambda_{i,j} : S_i \longrightarrow S_j$ ($i \leqslant j$ in I), there is a unique structure of an iterative monad on the functor

$$S = \operatorname*{colim}_{i \in I} S_i$$

turning the colimit morphisms into ideal monad homomorphisms.

Remark. The above colimit is, of course, formed objectwise: we define SX by $SX = \operatorname*{colim}_{i \in I} S_i X$ for all objects X in $\mathscr{A}$, and analogously for morphisms.

Proof. (I) We are given ideal monads (S_i, η_i, μ_i) with appropriate $\sigma_i : S_i' \longrightarrow S_i$ and $\mu_i' : S_i' S_i \longrightarrow S_i'$ for $i \in I$ and we form a colimit

$$(\lambda_i : S_i \longrightarrow S \mid i \in I)$$

in the functor category $[\mathscr{A}, \mathscr{A}]$. It is proved in Section 4 of Kelly and Power (1993) that there is a unique structure (S, η, μ) of a monad on the functor S such that all the transformations λ_i are monad homomorphisms, forming a colimit cocone in the category of all finitary monads on $\mathscr{A}$.

(II) (S, η, μ) is an ideal monad. In fact, we have a directed diagram of all S_i', $i \in I$ and all $\lambda_{i,j}' : S_i' \longrightarrow S_j'$, $i \leqslant j$, such that

$$S_i = S_i' + Id$$

and

$$\lambda_{i,j} = \lambda_{i,j}' + id$$

for all $i \leqslant j$ in I. (To show that $(\lambda_{i,j}')$ form a directed diagram, use the fact that $(\lambda_{i,j})$ do and that all σ_i are monomorphisms, being coproduct injections of $S_i = S_i' + Id$.) Put

$$S' = \operatorname*{colim}_{i \in I} S_i'$$

with the colimit cocone denoted by $\lambda_i' : S_i' \longrightarrow S'$, $i \in I$. Define $\sigma : S' \longrightarrow S$ as the unique natural transformation such that diagrams

$$\begin{array}{ccc} S' & \xrightarrow{\sigma} & S \\ {\scriptstyle \lambda_i'}\uparrow & & \uparrow{\scriptstyle \lambda_i} \\ S_i' & \xrightarrow[\sigma_i]{} & S_i \end{array}$$

commute for every $i \in I$. (This is, once more, forced on us by the requirement that λ_i is an ideal monad homomorphism.)

Since the diagonal functor $I \longrightarrow I \times I$ is cofinal, the composite $S'S$ is a colimit of the diagram of $S_i' S_i$, $i \in I$, with connecting morphisms

$$S_i' S_i \xrightarrow{S_i' \lambda_{i,j}} S_i' S_j \xrightarrow{\lambda_{i,j}' S_j} S_j' S_j \quad \text{for } i \leqslant j$$

and colimit cocone

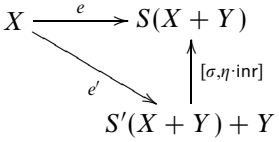
$$S_i' S_i \xrightarrow{S_i' \lambda_i} S_i' S \xrightarrow{\lambda_i' S} S'S \quad \text{for } i \in I.$$

Then $\mu' : S'S \longrightarrow S'$ is defined by commutativity of the squares

$$\begin{array}{ccc} S'S & \xrightarrow{\mu'} & S' \\ {\scriptstyle \lambda_i' S}\uparrow & & \uparrow \\ S_i' S & & {\scriptstyle \lambda_i'}\uparrow \\ {\scriptstyle S_i' \lambda_i}\uparrow & & \\ S_i' S_i & \xrightarrow[\mu_i']{} & S_i' \end{array}$$

for all $i \in I$. The verification that (S, η, μ) is an ideal monad is quite mechanical.

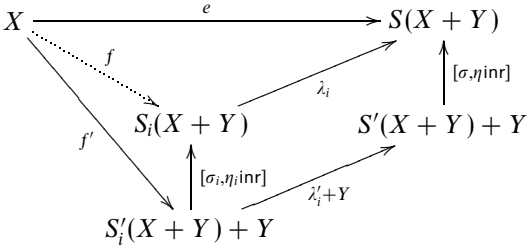
(III) (S, η, μ) is iterative. In fact, let



be a guarded equation morphism. Since X is a finitely presentable object, e' factors through one of the colimit morphisms of the filtered colimit

$$S'(X+Y)+Y = \operatorname*{colim}_{i\in I} S'_i(X+Y)+Y$$

with colimit cocone formed by $(\lambda'_i)_{X+Y} + Y$, $i \in I$. That is, we have $i \in I$ and f' such that the diagram



(47)

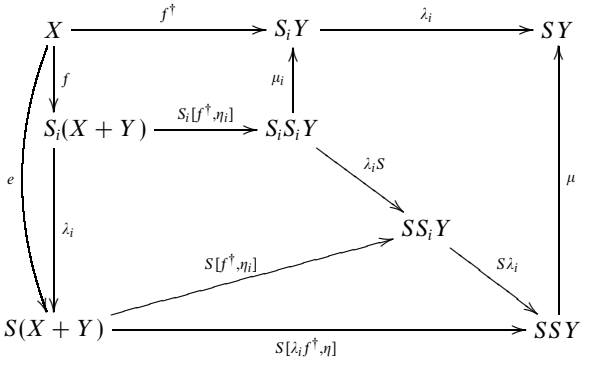
commutes. This defines a guarded equation morphism $f : X \longrightarrow S_i(X+Y)$ for S_i. We use $f^\dagger : X \longrightarrow S_i Y$ to denote the unique solution with respect to S_i. Then the composite

$$X \xrightarrow{f^\dagger} S_i Y \xrightarrow{\lambda_i} SY$$

is a solution of e with respect to S. In fact, the diagram (48) obviously commutes. It remains to show that solutions are unique for S. In fact, let

$$h : X \longrightarrow SY$$

be a solution of e. We prove $h = \lambda_i \cdot f^\dagger$. Observe first that the solution $\lambda_i \cdot f^\dagger$ is independent of the above factorisation $e' = ((\lambda'_i)_{X+Y} + Y) \cdot f'$. In fact, choose any other factorisation,

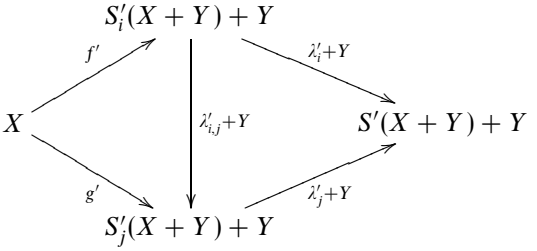


(48)

through $j \in I$:

$$e' = ((\lambda'_j)_{X+Y} + Y) \cdot g' \quad \text{for some } g' : X \longrightarrow S'_j(X+Y) + Y.$$

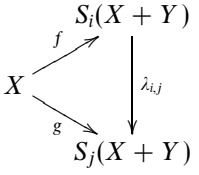
Since I is directed, we can assume the existence of some $(\lambda'_{i,j})_{X+Y} : S'_i(X+Y) \longrightarrow S'_j(X+Y)$ such that the diagram



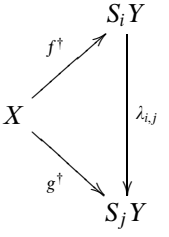
(49)

commutes.

Put $g = [(\sigma_j)_{X+Y}, (\eta_j)_{X+Y} \cdot \mathsf{inr}] \cdot g'$, analogously to (47). Notice that the triangle



commutes. By Proposition 5.3 applied to $\lambda = \lambda_{i,j}$, the triangle



commutes, where, $g^\dagger : X \longrightarrow S_jY$ denotes the unique solution with respect to S_j. This proves

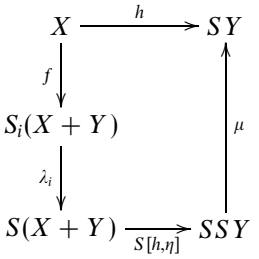
$$(\lambda_j)_Y \cdot g^\dagger = (\lambda_j)_Y \cdot (\lambda_{i,j})_Y f^\dagger = (\lambda_i)_Y \cdot f^\dagger, \tag{50}$$

as desired.

We are ready to prove that solutions for S are unique. Let

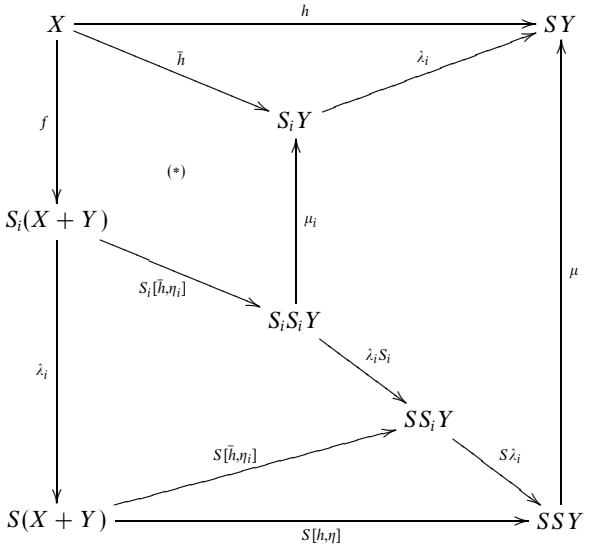
$$h : X \longrightarrow SY$$

be a solution, that is, let the diagram



(51)

commute. We can factor h through one of the colimit maps of the filtered colimit defining SY; without loss of generality we can assume that it is $(\lambda_i)_Y$ for the given $i \in I$ (recall the independence of $(\lambda_i)_Y \cdot f^\dagger$ of the given factorisation of e). We obtain the diagram (52):



(52)

The proof would be finished if we knew that $(*)$ in that diagram commutes: then $\bar{h} = f^\dagger$ implies $h = \lambda_i \cdot f^\dagger$, as required. However, we do not claim that commutativity. All we claim is that all the other inner parts of (52) commute (trivially) and the outer shape does by (51), thus,

$$(\lambda_i)_Y \text{ merges the two sides of } (*).$$

This implies, since the domain X of $(*)$ is finitely presentable and $(\lambda_i)_Y$ is a colimit morphism of the filtered colimit defining SY, that some connecting morphism $(\lambda_{i,j})_Y$ for $i \leqslant j$ in I also merges the two sides of $(*)$. Put

$$g' \equiv X \xrightarrow{f'} S'_i(X+Y) + Y \xrightarrow{\lambda'_{i,j}+Y} S'_j(X+Y) + Y$$

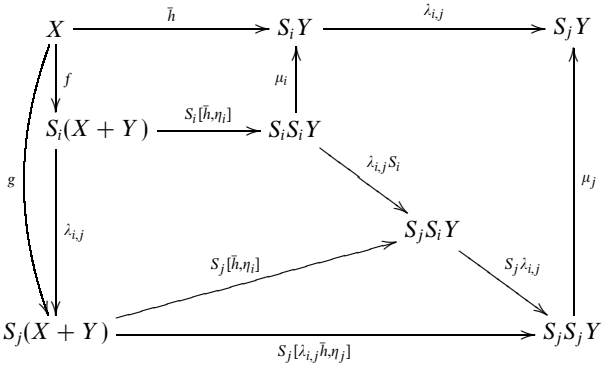
and

$$g \equiv X \xrightarrow{g'} S'_j(X+Y) + Y \xrightarrow{[\sigma_j, \eta_j \mathsf{inr}]} S_j(X+Y)$$

We claim that g, which obviously is a guarded equation morphism for S_j, has the solution

$$g^\dagger = (\lambda_{i,j})_Y \cdot \bar{h} : X \longrightarrow S_j Y. \tag{53}$$

To prove this, we observe that the diagram (54) commutes:



(54)

all the inner parts commute trivially except the upper left-hand square – and, by our choice of j, that square commutes when composed with $(\lambda_{i,j})_Y$. This proves (53). Therefore, we have, by (50),

$$(\lambda_i)_Y \cdot f^\dagger = (\lambda_j)_Y \cdot g^\dagger = (\lambda_j)_Y \cdot (\lambda_{i,j})_Y \bar{h} = (\lambda_i)_Y \bar{h} = h,$$

which concludes the proof. □

Definition 5.6. For every finitary functor H, expressed as a filtered colimit of strongly finitary functors H_i we define the *rational monad* R of H as the colimit of the corresponding diagram of the rational monads R_i of H_i,

$$R = \operatorname*{colim}_i R_i, \quad \text{with injections } \bar{\beta}_i : R_i \longrightarrow R$$

(see Remark 5.2).

We have yet to show that R is well-defined, that is, independent of the given representation of H as $H = \operatorname{colim} H_i$. This follows from the following proposition.

Proposition 5.7. The rational monad R of H is a free iterative monad on H. That is:
(a) R is an iterative monad;
(b) the natural transformation $\rho^* = \operatorname*{colim}_{i\in I} \rho_i^* : H \longrightarrow R$ is ideal;
and
(c) given an iterative monad S and an ideal natural transformation $\lambda : H \longrightarrow S$, there exists a unique ideal monad homomorphism $\bar{\lambda} : R \longrightarrow S$ with $\lambda = \bar{\lambda} \cdot \rho^*$.

Proof. For (a) see Proposition 5.5. Since each $\rho_i^* : H_i \longrightarrow R_i$ is an ideal transformation, that is, $\rho_i^* = \rho_i \cdot H\eta^{R_i}$, it follows that

$$\operatorname*{colim}_{i\in I} \rho_i^* = \Big(\operatorname*{colim}_{i\in I} \rho_i\Big) \cdot H\Big(\operatorname*{colim}_{i\in I} \eta^{R_i}\Big)$$

is also ideal. It remains to prove (c).

For each $i \in I$ we have an ideal transformation

$$H_i \xrightarrow{\beta_i} H \xrightarrow{\lambda} S,$$

which, by Theorem 4.30, yields a unique ideal monad homomorphism $\bar{\lambda}_i : R_i \longrightarrow S$ such that the square

$$\begin{array}{ccc} H_i & \xrightarrow{\rho_i^*} & R_i \\ {\scriptstyle \beta_i}\downarrow & & \downarrow{\scriptstyle \bar{\lambda}_i} \\ H & \xrightarrow[\lambda]{} & S \end{array} \tag{55}$$

commutes. The unicity makes it clear that $\bar{\lambda}_i$, $i \in I$, is a cocone of the diagram $(R_i \mid i \in I)$. Thus, we have a unique ideal monad homomorphism $\bar{\lambda} : R \longrightarrow S$ with

$$\bar{\lambda} \cdot \bar{\beta}_i = \bar{\lambda}_i \quad \text{for } i \in I. \tag{56}$$

Then, from $\rho^* \cdot \beta_i = \bar{\beta}_i \cdot \rho_i^*$ (recall $\rho^* = \operatorname{colim} \rho_i^*$), we obtain

$$\begin{aligned} (\bar{\lambda} \cdot \rho^*) \cdot \beta_i &= \lambda \cdot \bar{\beta}_i \cdot \rho_i^* \\ &= \bar{\lambda}_i \cdot \rho_i^* \qquad \text{by (56)} \\ &= \lambda \cdot \beta_i \qquad \text{by (55)} \end{aligned}$$

for all $i \in I$, which proves

$$\bar{\lambda} \cdot \rho^* = \lambda.$$

The unicity of $\bar{\lambda}$ is obvious: given an ideal homomorphism $\bar{\lambda}$ with $\bar{\lambda} \cdot \rho^* = \lambda$, we have $\bar{\lambda} \cdot \bar{\beta}_i : R_i \longrightarrow S$ is an ideal homomorphism with

$$\bar{\lambda} \cdot \bar{\beta}_i \cdot \rho_i^* = \bar{\lambda} \cdot \rho^* \cdot \beta_i = \lambda \cdot \beta_i,$$

which (since R_i is free on H_i) determines $\bar{\lambda} \cdot \bar{\beta}_i$ for $i \in I$, and this determines $\bar{\lambda}$. □

Recall that guarded equation morphisms $e : X \longrightarrow R(X+Y)$ with X finitely presentable are called rational.

Corollary 5.8. (Rational Solution Theorem) Every rational equation morphism $e : X \longrightarrow R(X + Y)$ has a unique solution. That is, there exists a unique morphism

$$e^\ddagger : X \longrightarrow RY$$

such that the triangle

$$\begin{array}{ccc} X & \xrightarrow{e^\ddagger} & RY \\ {\scriptstyle e}\downarrow & \nearrow_{\widehat{[e^\ddagger, \eta_Y^R]}} & \\ R(X+Y) & & \end{array}$$

commutes (where for $s = [e^\ddagger, \eta_Y^R]$ we denote $\widehat{s} = \mu_Y^R \cdot Rs$, as always).

In fact, if Y is finitely presentable, this follows immediately from Proposition 5.7(a). If Y is arbitrary, we express it as a filtered colimit of finitely presentable objects $Y = \operatorname*{colim}_{i \in I} Y_i$,

with a colimit cocone $y_i : Y_i \longrightarrow Y$. Since R is finitary, we obtain $R(X+Y) = \operatorname{colim}_{i\in I} R(X+Y_i)$. By the finite presentability of X, e factors through one of the colimit morphisms $R(X+y_i)$ through a guarded equation morphism $e_i : X \longrightarrow R(X+Y_i)$. It is a routine verification that the existence and uniqueness of the solution of e_i implies the existence and uniqueness of the solution of e.

Remark 5.9. The rational monad R fulfills

$$R = HR + Id$$

with coproduct injections

$$\rho : HR \longrightarrow R$$

(turning each object RY into an H-algebra) given by

$$\rho = \operatorname*{colim}_i \rho_i : \operatorname*{colim}_i H_i R_i \longrightarrow \operatorname*{colim}_i R_i$$

and

$$\eta : Id \longrightarrow R$$

(the unit of the monad).

In fact, filtered colimits commute with finite coproducts, so all this follows from $R_i = H_i R_i + Id$ with coproduct injections ρ_i and η_i.

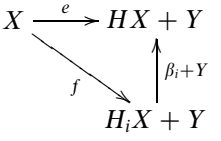
Remark 5.10. We are going to establish the formulas $RY = \bigcup \mathrm{im}(e^\dagger)$ and $RY = \operatorname{colim} \mathrm{Eq}_Y$ for all objects Y of $\mathscr{A}$ (cf. 4.2. and 4.6.). It is sufficient to prove the second one for all finitely presentable objects Y, the extension to all objects is exactly as in Remark 4.20.

For Y in $\mathscr{A}_{fp}$, the diagram $\mathrm{Eq}_Y^{(i)}$ of all finitary flat equations $e : X \longrightarrow H_i X + Y$ (X finitely presentable) has a colimit

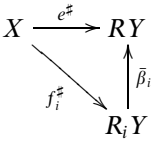
$$R_i Y = \operatorname{colim} \mathrm{Eq}_Y^{(i)}$$

with colimit cocone $e_i^\sharp : X \longrightarrow R_i Y$, see Section 4.

Consider an arbitrary finitary flat equation morphism $e : X \longrightarrow HX + Y$ for H. Since X is finitely presentable and $HX + Y$ is a directed colimit of $H_i X + Y$, $i \in I$, we have a factorisation as follows:



for some $i \in I$. Here f is a finitary flat equation morphism for H_i, and we use $f_i^\sharp : X \longrightarrow R_i Y$ to denote the corresponding colimit map of $R_i Y = \operatorname{colim} \mathrm{Eq}_Y^{(i)}$. Define $e^\sharp$ by



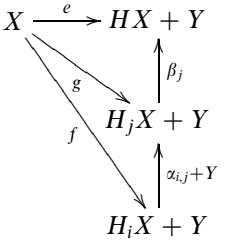
This is well-defined (that is, independent of the factorisation) and forms a colimit of the diagram Eq_Y, as we prove now.

Proposition 5.11. For every finitely presentable object Y we have

$$RY = \operatorname{colim} \mathrm{Eq}_Y$$

with the above colimit cocone $e^\sharp : X \longrightarrow RY$ (e in EQ_Y).

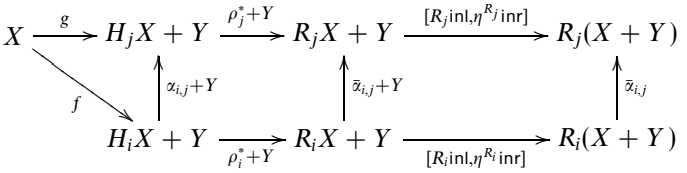
Proof. (I) **Independence of the factorisation:** Since I is a directed poset, all we have to prove is that given $i \leqslant j$ in I, and, using $g : X \longrightarrow H_jX + Y$ to denote the composite of f and $(\alpha_{i,j})_X + Y$,



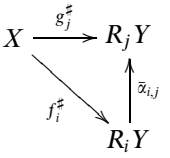
we have

$$\bar{\beta}_i \cdot f_i^\sharp = \bar{\beta}_j \cdot g_j^\sharp.$$

Since the diagram

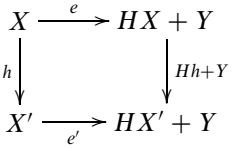


commutes, it follows from Corollary 5.4 that the solutions $f_i^\sharp : X \longrightarrow R_iY$ (with respect to R_i) and $g_j^\sharp : X \longrightarrow R_jY$ (with respect to R_j) also form a commutative triangle:



Combined with the fact that $\bar{\beta}_i = \bar{\beta}_j \cdot \bar{\alpha}_{i,j}$ (because $\beta_i = \beta_j \cdot \alpha_{i,j}$), this yields the desired equality $\bar{\beta}_i \cdot f_i^\sharp = \bar{\beta}_j \cdot g_j^\sharp$.

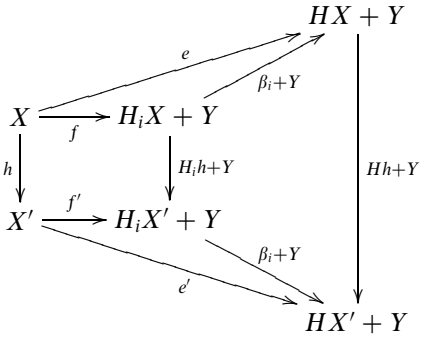
(II) **The morphisms $e^\sharp$ form a cocone of Eq_Y:** That is, given a morphism



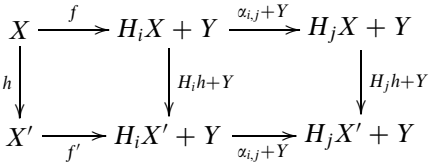
in EQ_Y, we have $e^\sharp = (e')^\sharp \cdot h$ holds. Since X and X' are finitely presentable, there exists $i \in I$ such that e factors as $e = ((\beta_i)_X + Y) \cdot f$ and e' as $e' = ((\beta_i)_{X'} + Y) \cdot f'$. Next, observe that

$$HX + Y = \operatorname*{colim}_{i \in I} H_i X + Y$$

is a filtered colimit with colimit cocone $(\beta_i)_X + Y$. The parallel pair $f' \cdot h, (H_i h + Y) \cdot f : X \longrightarrow H_i X' + Y$ gets merged by the colimit morphism $(\beta_i)_{X'} + Y$:



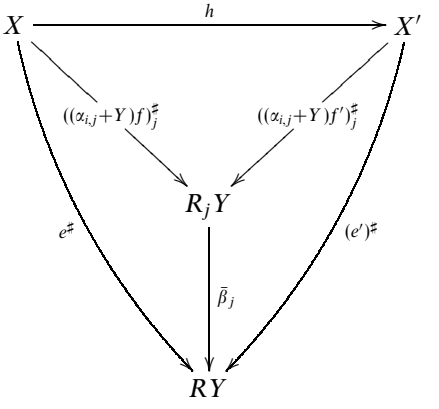
Since the domain X of that parallel pair is finitely presentable, it follows that some connecting morphism $(\alpha_{i,j})_{X'} + Y$ of the above filtered diagram also merges that parallel pair. That is, in the diagram



the outward square commutes. Consequently, we have a morphism h in the filtered diagram defining $R_j Y$, therefore,

$$(((\alpha_{i,j})_X + Y) \cdot f)^\sharp_j = (((\alpha_{i,j})_{X'} + Y) \cdot f')^\sharp_j \cdot h.$$

This proves that the desired triangle



commutes, since $e^\sharp$ and $(e')^\sharp$ are by (I) independent of the above factorisation.

(III) **Universal property.** Let a cocone $e^{@} : X \longrightarrow C$ (for e in EQ_Y) of the diagram Eq_Y be given. For each $i \in I$, we obtain a cocone of $D_Y^{(i)}$ (the diagram of all finitary flat equation morphisms of H_i with respect to Y) as follows: to every $f : X \longrightarrow H_iX + Y$, we assign

$$(((\beta_i)_X + Y) \cdot f)^{@} : X \longrightarrow C.$$

This is indeed a cocone, since for every morphism

$$\begin{array}{ccc} X & \xrightarrow{f} & H_iX + Y \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle H_ih+Y} \\ X' & \xrightarrow[f']{} & H_iX' + Y \end{array}$$

in $\mathsf{EQ}_Y^{(i)}$, we have a morphism

$$\begin{array}{ccccc} X & \xrightarrow{f} & H_iX + Y & \xrightarrow{\beta_i+Y} & HX + Y \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle H_ih+Y} & & \downarrow{\scriptstyle Hh+Y} \\ X' & \xrightarrow[f']{} & H_iX' + Y & \xrightarrow[\beta_i+Y]{} & HX' + Y \end{array}$$

in EQ_Y, thus, $(((\beta_i)_X + Y) \cdot f)^{@} = (((\beta_i)_{X'} + Y) \cdot f')^{@} \cdot h$. Therefore, we get a unique

$$c_i : R_iY \longrightarrow C$$

with

$$c_i \cdot f_i^{\sharp} = (((\beta_i)_X + Y) \cdot f)^{@} \quad \text{for all } f \text{ in } \mathsf{EQ}_Y^{(i)}.$$

The uniqueness of c_i, for each $i \in I$, makes it obvious that these morphisms form a cocone of the diagram $(R_iY \mid i \in I)$. Thus, there is a unique

$$c : RY \longrightarrow C$$

with

$$c \cdot (\bar{\beta}_i)_Y = c_i \quad \text{for all } i \in I.$$

This morphism fulfills

$$c \cdot e^{\sharp} = e^{@} \quad \text{for each } e \text{ in } \mathsf{EQ}_Y.$$

In fact, we can factor e via $f : X \longrightarrow H_iX + Y$ as above, and then

$$\begin{aligned} c \cdot e^{\sharp} &= c \cdot (\bar{\beta}_i)_Y \cdot f_i^{\sharp} && \text{by the definition of } e^{\sharp} \\ &= c_i \cdot f_i^{\sharp} && \text{by the definition of } c \\ &= (((\beta_i)_X + Y) \cdot f)^{@} && \text{by the definition of } c_i \\ &= e^{@}. \end{aligned}$$

To prove that c is unique, we only have to observe that given a morphism $c : RY \longrightarrow C$ with $c \cdot e^{\sharp} = e^{@}$ for each e in EQ_Y, it follows that $c \cdot (\bar{\beta}_i)_Y = c_i$ for each $i \in I$ (which determines c). In fact, R_iY is a colimit of $\mathrm{Eq}_Y^{(i)}$ with colimit morphisms $f_i^{\sharp}$, and from

$c \cdot e^\sharp = e^{@}$, we conclude, for each $f : X \longrightarrow H_i X + Y$ in $\mathsf{EQ}_Y^{(i)}$, that

$$\begin{aligned} c \cdot (\bar{\beta}_i)_Y \cdot f_i^\sharp &= c \cdot (((\beta_i)_X + Y) \cdot f)^\sharp \\ &= (((\beta_i)_X + Y) \cdot f)^{@} \\ &= c_i \cdot f_i^\sharp \end{aligned}$$

□

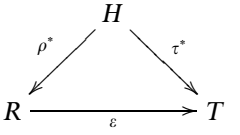
Remark 5.12. Recall that since H is iteratable, it generates the monad T of Corollary 1.8. Together with

$$\tau^* \equiv H \xrightarrow{H\eta} HT \xrightarrow{\tau} T,$$

this is a free completely iterative monad on H. We use

$$\varepsilon : R \longrightarrow T$$

to denote the unique ideal monad homomorphism with



commutative, see Proposition 5.7. By Proposition 5.3, ε preserves solutions, thus, it satisfies (due to Remark 4.27)

$$\varepsilon_Y \cdot e^\sharp = e^\dagger \quad \text{for all } e : X \longrightarrow HX + Y \text{ in } \mathsf{EQ}_Y,$$

for all objects Y of $\mathscr{A}$. Since RY is the union of all images of finitary flat equation morphisms (see Remark 5.10), it follows that ε_Y is a monomorphism (*viz.*, the monomorphism representing that union). This proves the following corollary.

Corollary 5.13. R is a submonad of the monad T via ε.

Theorem 5.14 (Rational Substitution Theorem). For every morphism

$$s : X \longrightarrow RY,$$

there exists a unique extension to a homomorphism

$$\widetilde{s} : RX \longrightarrow RY$$

of H-algebras (that is, a unique homomorphism with $s = \widetilde{s} \cdot \eta_Y^R$).

Proof. Existence is clear: $\widetilde{s} = \mu_Y \cdot Rs$. Uniqueness is proved precisely as in Theorem 4.16. □

6. The rational monad of a set functor

We are going to define a rational monad of every finitary functor H of Set. In the last section, this has been done whenever H preserves monomorphisms.

Now all monomorphisms $m : X \longrightarrow Y$ in Set with $X \neq \emptyset$ are split – choose $x_0 \in X$ and define $e : Y \longrightarrow X$ by

$$e(y) = \begin{cases} x, & \text{if } m(x) = y \\ x_0, & \text{otherwise.} \end{cases}$$

Then $em = id$. Thus, every functor preserves these monomorphisms. The only trouble-makers are, thus, the empty functions

$$\psi_X : \emptyset \longrightarrow X \quad (X \neq \emptyset).$$

Not all finitary functors preserve monomorphisms. For example, put $HX = 1$ (terminal object) for $X \neq \emptyset$ and $H\emptyset = 1 + 1$. This defines H uniquely, and $H\psi_1 : 1 + 1 \longrightarrow 1$ is, of course, no monomorphism. Luckily, iterative monads in Set do preserve monomorphisms.

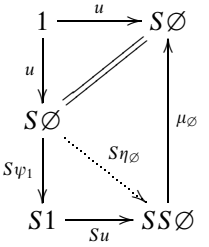
Proposition 6.1. Every iterative monad in Set preserves monomorphisms.

Proof. Let S be an iterative monad. Observe that $S\emptyset = S'\emptyset$ (since $S = S' + Id$).

If $S'\emptyset = \emptyset$, then $S\psi_X : \emptyset \longrightarrow SX$ is a monomorphism and the proof is concluded. Suppose $S'\emptyset \neq \emptyset$, choose $u' : 1 \longrightarrow S'\emptyset$ and put $u = \sigma_\emptyset \cdot u' : 1 \longrightarrow S\emptyset$. Then the composite

$$e \equiv 1 \xrightarrow{u} S\emptyset \xrightarrow{S\psi_1} S1$$

is a guarded equation morphism for the empty set $Y = \emptyset$ of parameters. And u is a solution of e, that is, the diagram



(where $u = [u, \eta_\emptyset] : 1 + \emptyset \longrightarrow S\emptyset$) commutes. In fact, we have $u \cdot \psi_1 = \eta_\emptyset : \emptyset \longrightarrow S\emptyset$ because $\mathsf{Set}(\emptyset, S\emptyset)$ is a singleton set; the rest is clear. This proves that $S\psi_1$ is a monomorphism: given $u, v : 1 \longrightarrow S\emptyset$ with $S\psi_1 \cdot u = S\psi_1 \cdot v = e$, then both u and v are solutions of e, and thus, $u = v$. Consequently, $S\psi_X$ is a monomorphism for every X. □

Remark 6.2. For every endofunctor H of Set, the equivalence relation on $H\emptyset$ given by

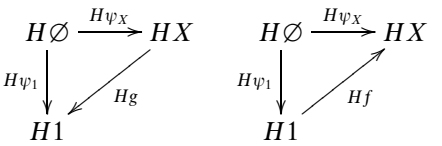
$$u \sim v \quad \text{iff} \quad H\psi_X(u) = H\psi_X(v)$$

where X is a fixed non-empty set, is independent of the choice of X.

In brief,

$$H\psi_X(u) = H\psi_X(v) \quad \text{iff} \quad H\psi_1(u) = H\psi_1(v).$$

In fact, since $X \neq \emptyset$, we have functions $f : 1 \longrightarrow X$ and $g : X \longrightarrow 1$, and the triangles



commute (because $\emptyset$ is the initial object).

Definition 6.3. For every endofunctor H of Set, we define a quotient

$$\gamma : H \longrightarrow H^+$$

of H as follows.

For all non-empty sets X, we put $H^+X = HX$ and $\gamma_X = id$; for the empty set, let

$$\gamma_\emptyset : H\emptyset \longrightarrow H\emptyset/\sim = H^+\emptyset$$

be the canonical function of the equivalence $\sim$ of Remark 6.2.

For all functions $f : X \longrightarrow Y$ with $X \neq \emptyset$, put $H^+f = Hf$, and for the empty function put $H^+id_\emptyset = id_{H^+\emptyset}$ and

$$H^+\psi_X : [u] \mapsto H\psi_X(u) \quad \text{for all } u \in H\emptyset$$

if X is non-empty.

The above definition is a small modification of a procedure used by V. Trnková (Trnková 1971).

We now prove that H^+ is a reflection of H in the category of all endofunctors of Set preserving monomorphisms.

Lemma 6.4. For every endofunctor H of Set, the functor H^+ preserves monomorphisms, and $\gamma : H \longrightarrow H^+$ has the following universal property:

> For every natural transformation $\delta : H \longrightarrow K$, where K preserves monomorphisms, there exists a unique natural transformation $\delta^+ : H^+ \longrightarrow K$ with $\delta = \delta^+\gamma$.

Proof. (I) H^+ preserves monomorphisms (equivalently, it maps ψ_1 to a monomorphism), since $[u] \neq [v]$ holds iff $H\psi_1(u) \neq H\psi_1(v)$, for all $u, v \in H\emptyset$.

(II) Let $\delta : H \longrightarrow K$ be given. The equation $\delta = \delta^+\gamma$ forces us to define

$$\delta^+_X = \delta_X : HX \longrightarrow KX \quad \text{for all } X \neq \emptyset.$$

It remains to discuss $\delta^+_\emptyset : H\emptyset/\sim \longrightarrow K\emptyset$. All we have to prove is that for $u, v \in H\emptyset$, we have

$$u \sim v \quad \text{implies} \quad \delta_\emptyset(u) = \delta_\emptyset(v). \tag{57}$$

Then $\delta^+_\emptyset$ is uniquely determined by $\delta_\emptyset = \delta^+_\emptyset \cdot \gamma_\emptyset$ (*viz.*, $\delta^+_\emptyset([u]) = \delta_\emptyset(u)$), and the naturality of δ^+ is obvious. To show (57), use naturality of δ on ψ_1:

$$\begin{array}{ccc} H\emptyset & \xrightarrow{\delta_\emptyset} & K\emptyset \\ {\scriptstyle H\psi_1}\downarrow & & \downarrow{\scriptstyle K\psi_1} \\ H1 & \xrightarrow[\delta_1]{} & K1 \end{array}$$

From $u \sim v$ we conclude, since $H\psi_1(u) = H\psi_1(v)$, that $K\psi_1(\delta_\emptyset(u)) = K\psi_1(\delta_\emptyset(v))$. Since $K\psi_1$ is a monomorphism, it follows that $\delta_\emptyset(u) = \delta_\emptyset(v)$. □

Definition 6.5. For every finitary endofunctor H the *rational monad of* H is defined to be the rational monad $\rho^* : H^+ \longrightarrow R$ of the above reflection $\gamma : H \longrightarrow H^+$.

Corollary 6.6. The rational monad of H, together with the natural transformation

$$H \xrightarrow{\gamma} H^+ \xrightarrow{\rho^*} R$$

is a free iterative monad on H.

Proof. In fact, since ρ^* is an ideal natural transformation, so is $\rho^*\gamma$. Let S be an iterative monad and $\lambda : H \longrightarrow S$ be an ideal transformation, $\lambda = \sigma\lambda'$. Since the subfunction S' of S preserves monomorphisms (see Proposition 6.1), the natural transformation $\lambda' : H \longrightarrow S'$ extends uniquely to $(\lambda')^+ : H^+ \longrightarrow S'$, and we obtain an ideal natural transformation

$$\lambda^+ \equiv H^+ \xrightarrow{(\lambda')^+} S' \xrightarrow{\sigma} S.$$

By Proposition 5.7, this yields a unique ideal monad homomorphism $\overline{\lambda^+} : R \longrightarrow S$ extending λ^+, that is, such that

$$\begin{array}{ccccc} H & \xrightarrow{\gamma} & H^+ & \xrightarrow{\rho^*} & R \\ & \searrow^{\lambda} & & \searrow^{\lambda^+} & \downarrow{\scriptstyle \overline{\lambda^+}} \\ & & & & S \end{array}$$

commutes. □

7. Conclusion and future directions

We have constructed a free iterative monad on every monos-preserving, finitary endofunctor H of a 'set-like' category. In particular, every finitary endofunctor of Set generates a free iterative monad (or free iterative Lawvere theory). The method of our construction was coalgebraic, making heavy use of the fact that final coalgebras TX of the functors $H(\_) + X$ form a free completely iterative monad on H.

The proof is surprisingly technical, and one question it naturally raises is whether a simpler proof can be found. When H is a polynomial functor, the existence of a free iterative theory was established by C. C. Elgot (Elgot 1975) and the fact that this

is the theory of rational trees was proved later in Elgot *et al.* (1978). Their method is fundamentally different from ours, and their proof was also quite involved.

The rational monad as presented here is based on solutions of finitary flat equations. This can be extended to solutions of finitary equations in much the same style as mentioned, for polynomial functors, in the Introduction (see Section 1.1), we have proved this is Adámek *et al.* (2002).

In the work of Bloom and Ésik (Bloom and Ésik 1993), equational properties of solutions are described and revealed to have a general pattern in various fields. See also a recent restatement of some of these results in Simpson and Plotkin (2000). It would be interesting to investigate these properties categorically. L. Moss (Moss 2001) has already taken first steps in that direction.

Acknowledgements

The present paper has evolved out of a very pleasant cooperation between the authors and Peter Aczel. In the joint paper Aczel *et al.* (2002), whose extended abstract was published as Aczel *et al.* (2001), we constructed a free completely iterative monad on every iteratable functor. The current paper has also been discussed with Peter, and the inspiration obtained from him is gratefully acknowledged. We are also grateful to the referees for their suggestions for improvements of presentation.

References

Aczel, P., Adámek, J. and Velebil, J. (2001) A coalgebraic view of infinite trees and iteration. *Electronic Notes in Theoretical Computer Science* **44** (1).

Aczel, P., Adámek, J., Milius, S. and Velebil, J. (2002) Infinite trees and completely iterative theories: A Coalgebraic View. (To appear in *Theoretical Computer Science*.)

Adámek, J. (1997) A categorical generalisation of Scott domains. *Mathematical Structures in Computer Science* **7** 419–443.

Adámek, J. and Koubek, V. (1995) On the greatest fixed point of a set functor. *Theoretical Computer Science* **150** 57–75.

Adámek, J., Milius, S. and Velebil, J. (2002) Some remarks on finitary and iterative monads (submitted).

Adámek, J. and Rosický, J. (1994) *Locally Presentable and Accessible Categories*, Cambridge University Press.

Adámek, J. and Trnková, V. (1990) *Automata and Algebras in Categories*, Kluwer Academic Publishers.

Bloom, S. L. and Ésik, Z. (1993) *Iteration Theories: The Equational Logic of Iterative Processes*, EATCS Monograph Series on Theoretical Computer Science, Springer-Verlag.

Carboni, A., Lack, S. and Walters, R. F. C. (1993) Introduction to extensive and distributive Categories. *J. Pure Appl. Algebra* **84** 145–158.

Elgot, C. C. (1975) Monadic computation and iterative algebraic theories. In: Rose, H. E. and Shepherdson, J. C. (eds.) *Logic Colloquium '73*, North-Holland 175–230.

Elgot, C. C., Bloom, S. L. and Tindell, R. (1978) On the algebraic structure of rooted trees. *J. Comp. Syst. Sciences* **16** 361–399.

Gabriel, P. and Ulmer, F. (1971) Lokal präsentierbare Kategorien. *Springer-Verlag Lecture Notes in Mathematics* **221**.

Kelly, M. and Power, J. (1993) Adjunctions whose counits are coequalisers, and presentations of finitary enriched monads. *J. Pure Appl. Algebra* **89** 163–179.

Lambek, J. (1968) A fixpoint theorem for complete categories. *Math. Z.* **103** 151–161.

Manes, E. G. (1976) *Algebraic Theories*, Springer-Verlag.

Moss, L. (2001) Parametric corecursion. *Theoretical Computer Science* **260** (1-2) 139–163.

Moss, L. (2000) Recursion and corecursion have the same equational logic. (To appear in *Theorectical Computer Science*.)

Simpson, A. and Plotkin, G. (2000) Complete axioms for categorical fixed-point operators. *Fifteenth Annual IEEE Symposium on Logic in Computer Science* 30–41.

Trnková, V. On descriptive classification of set functors. *Comment. Math. Univ. Carolinae* **12** 143–175.

FROM ITERATIVE ALGEBRAS TO ITERATIVE THEORIES

JIŘÍ ADÁMEK, STEFAN MILIUS, AND JIŘÍ VELEBIL

Abstract. Iterative theories introduced by Calvin Elgot formalize potentially infinite computations as unique solutions of recursive equations. One of the main results of Elgot and his coauthors is a description of a free iterative theory as the theory of all rational trees. Their algebraic proof of this fact is extremely complicated. In our paper we show that by starting with "iterative algebras", i. e., algebras admitting a unique solution of all systems of flat recursive equations, a free iterative theory is obtained as the theory of free iterative algebras. The (coalgebraic) proof we present is dramatically simpler than the original algebraic one. And our result is, nevertheless, much more general: we describe a free iterative theory on any finitary endofunctor of every locally presentable category $\mathcal{A}$. This allows us, e. g., to consider iterative algebras over every equationally specified class $\mathcal{A}$ of finitary algebras.

Reportedly, a blow from the welterweight boxer Norman Selby, also known as *Kid McCoy*, left one victim proclaiming, "*It's the real McCoy!*"

[TPT]

1. Introduction

Iterative theories have been introduced by Calvin C. Elgot [E] as a model of computation given by a sequence of instantaneous descriptions of an abstract machine. He and his co-authors then proved that for every signature Σ a free iterative theory on Σ exists [BE] and that it consists of all rational Σ-trees [EBT]. Recall that a Σ-tree (i. e., a tree, possibly infinite, labelled by operation symbols in Σ so that every node with n children is labelled by an n-ary symbol) is *rational* if it has up to isomorphism only finitely many subtrees, see [G].

In the present paper we introduce *iterative algebras* rather than iterative theories, and we show that the theory formed by all free iterative algebras is Elgot's free iterative theory. In the classical case of Σ-algebras, iterativity has been introduced by Evelyn Nelson [N] as follows: given a Σ-algebra A, let us consider an arbitrary system of recursive equations

$$x_i \approx t_i, \qquad i = 1, \dots, n\,, \tag{1.1}$$

where $X = \{x_1, x_2, \dots, x_n\}$ is a finite set of variables and t_1, t_2, …, t_n are terms over $X + A$, none of which is a single variable x_i. The algebra A is called *iterative* provided that for every such system of equations there exists a unique *solution*. That is, there exists a unique n-tuple ${x_1}^\dagger$, ${x_2}^\dagger$, …, ${x_n}^\dagger$ of elements of A such that each of the formal equations in (1.1) becomes an equality after the substitution ${x_i}^\dagger/x_i$:

$${x_i}^\dagger = t_i({x_1}^\dagger/x_1, {x_2}^\dagger/x_2, \dots, {x_n}^\dagger/x_n)\,, \qquad i = 1, \dots, n.$$

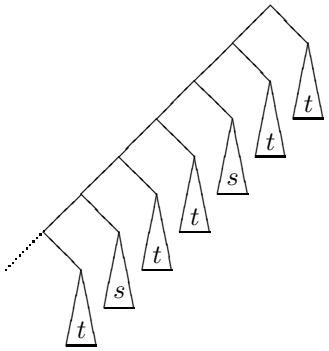
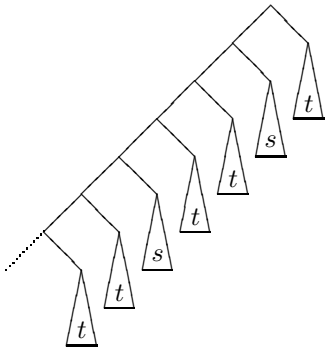
Example: let Σ consist of a single binary operation symbol, $*$, then the algebra A of all (finite and infinite) binary trees is iterative. For example, the system

$$\begin{array}{lcl} x_1 & \approx & x_2 * t \\ x_2 & \approx & (x_1 * s) * t \end{array} \tag{1.2}$$

Date: October 21, 2004.

The first and the third author acknowledge the support of the Grant Agency of the Czech Republic under the Grant No. 201/02/0148.

where s and t are trees in A has the unique solution

$x_1{}^\dagger =$  $x_2{}^\dagger =$  (1.3)

Every system (1.1) above can be modified to a *flat system*, i.e., one where each right-hand side is either a *flat term*

$$t_i = \sigma(y_1, \ldots, y_k), \quad \text{for } \sigma \in \Sigma_k \text{ and } y_1, \ldots, y_k \in X,$$

or an element of A

$$t_i \in A.$$

For example, the above system (1.2) has the following modification to a flat system:

$$\begin{array}{llllllll} x_1 & \approx & x_2 * x_3 & x_3 & \approx & t & x_5 & \approx & s \\ x_2 & \approx & x_4 * x_3 & x_4 & \approx & x_1 * x_5 & & & \end{array}$$

Therefore, an algebra is iterative iff every flat equation system has a unique solution.

Now Σ-algebras are a special case of algebras for an endofunctor $H : \mathcal{A} \longrightarrow \mathcal{A}$ (which are pairs consisting of an object A of $\mathcal{A}$ and a morphism $\alpha : HA \longrightarrow A$): here $\mathcal{A}$ is the category of sets and $H = H_\Sigma$ is the *polynomial functor* given on objects by

$$H_\Sigma X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X^2 + \cdots$$

For a Σ-algebra (A, α) observe that a flat equation system has its right-hand sides in $H_\Sigma X + A$, thus, it can be represented by a morphism

$$e : X \longrightarrow H_\Sigma X + A, \quad e(x_i) = t_i.$$

A *solution* of e is then a morphism

$$e^\dagger : X \longrightarrow A, \quad e^\dagger(x_i) = x_i{}^\dagger,$$

with the property that the following diagram

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha, A]} \\ H_\Sigma X + A & \xrightarrow[H_\Sigma e^\dagger + A]{} & H_\Sigma A + A \end{array} \tag{1.4}$$

commutes. This leads to the following definition concerning H-algebras for any endofunctor H of Set:

Definition 1.1. An H-algebra (A, α) is called *iterative* provided that for every flat equation morphism $e : X \longrightarrow HX + A$, where X is a finite set, there exists a unique solution, i. e., a unique morphism $e^\dagger : X \longrightarrow A$ such that the square

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha, A]} \\ HX + A & \xrightarrow[He^\dagger + A]{} & HA + A \end{array}$$

commutes.

"Classical" algebras are seldom iterative. But there are enough interesting iterative algebras. For example, the Σ-algebra

$$T_\Sigma$$

of all (finite and infinite) Σ-trees is iterative. And so is its subalgebra

$$R_\Sigma$$

of all rational Σ-trees. In fact, the full subcategory $\mathsf{Alg}_{it}\ \Sigma$ of $\mathsf{Alg}\ \Sigma$ formed by all iterative Σ-algebras is rich enough: a limit or a filtered colimit of iterative algebras is always iterative, thus $\mathsf{Alg}_{it}\ \Sigma$ is reflective in $\mathsf{Alg}\ \Sigma$. From this it follows that every set generates a free iterative algebra, i. e., the forgetful functor $\mathsf{Alg}_{it}\ \Sigma \longrightarrow \mathsf{Set}$ is a right-adjoint. This defines a monad $\mathbb{R}_\Sigma$ on Set. We prove that

(i) $\mathbb{R}_\Sigma$ is a free iterative monad on H_Σ,

and

(ii) $\mathbb{R}_\Sigma$ assigns to every set X the algebra $R_\Sigma X$ of all rational Σ-trees on X, i. e., rational trees where leaves are labelled by constant symbols from Σ or elements of X.

In this way a new proof of the result of Elgot et al. describing a free iterative monad (or theory) is achieved.

In our proof we work with an arbitrary endofunctor H of the category of sets which is *finitary*, i. e., preserves filtered colimits. The main technical result is coalgebraic: in order to describe a free iterative algebra on a set Y, we form the diagram Eq_Y of all coalgebras $e : X \longrightarrow HX + Y$ of the endofunctor $H(-) + Y$ on finite sets X. We prove that a colimit of that diagram

$$RY = \operatorname{colim} \mathrm{Eq}_Y$$

carries naturally the structure of an algebra, and that RY is a free iterative H-algebra on Y. From that we derive that the monad $R(-)$ is a free iterative monad on H. In our proof the fact that H is a finitary endofunctor of Set plays no rôle: the same result holds for finitary endofunctors of all locally finitely presentable categories. Thus, if we start e. g. with an equational class $\mathcal{A}$ of finitary algebras then, again, for every finitary endofunctor H the free iterative algebras RY are constructed as colimits of coalgebras of $H(-) + Y$ on finitely presentable objects of $\mathcal{A}$, and they form a free iterative theory on H.

Related Work. In the classical setting, i. e., for polynomial endofunctors of Set, iterative algebras were introduced by Evelyn Nelson [N] to obtain a short proof of Elgot's free iterative theories. Our paper can be seen as a categorical generalization of that paper with distinctive coalgebraic "flavour". Also Jerzy Tiuryn introduced a concept of iterative algebra in [T] with the same aim as ours: to relate iterative theories of Elgot to properties of algebras. But the approach of [T] is different from ours; e.g., the trivial, one-element, algebra is not iterative in the sense of Tiuryn, thus, his iterative algebras are not closed under limits.

The description of the rational monad as a colimit is also presented in [GLM].

The present paper is a dramatic improvement of our previous description of the rational monad in [AMV_1], [AMV_2] where we assumed that the endofunctor preserves monomorphisms and the underlying category satisfies three rather technical conditions, and the proof was much more involved. The current approach includes all equationally defined algebraic categories as base categories (whereas in [AMV_2] we still needed strong side conditions which only hold in very few algebraic categories). We believe that with this paper we have the "real McCoy". Simultaneously to the present paper the paper [Mi] devoted to completely iterative algebras evolved.

2. Iterative Algebras

Notation 2.1. Throughout the paper all categories are assumed to have finite coproducts. We denote by inl and inr the coproduct injections of $A + B$. For an endofunctor H, let $\mathsf{can} : HA + HB \longrightarrow H(A + B)$ denote the canonical morphism $\mathsf{can} = [H\mathsf{inl}, H\mathsf{inr}]$.

In order to define the concept of a flat equation morphism as in the introduction (a morphism $e : X \longrightarrow HX + A$ in Set where X is finite) in a general category, we need the appropriate generalization of finiteness. Recall that a functor is called *finitary* provided that it preserves filtered colimits. A set is finite if and only if its hom-functor is finitary. This has inspired Gabriel and Ulmer [GU] to the following

Definition 2.2. An object of A a category $\mathcal{A}$ is *finitely presentable* if its hom-functor $\mathcal{A}(A, -) : \mathcal{A} \longrightarrow \mathsf{Set}$ is finitary.

A category $\mathcal{A}$ is called *locally finitely presentable* provided that it has colimits and a (small) set of finitely presentable objects whose closure under filtered colimits is all of $\mathcal{A}$.

Examples 2.3.

(1) A poset is finitely presentable in Pos, the category of posets and order-preserving functions, if and only if it is finite. Pos is a locally finitely presentable category.
(2) The category CPO of complete partial orders and continuous functions is not locally finitely presentable: it has no nontrivial finitely presentable objects.

(3) Every variety of finitary algebras is locally finitely presentable. The categorical concept of finitely presentable object coincides with the algebraic one (of having finitely many generators and finitely many presenting equations), see [AR].
(4) Let H be a finitary endofunctor of a locally finitely presentable category $\mathcal{A}$. Then the category $\mathsf{Alg}\ H$ of H-algebras and homomorphisms is also locally finitely presentable, see [AR].

Definition 2.4. Given an endofunctor $H : \mathcal{A} \longrightarrow \mathcal{A}$, by a *finitary flat equation morphism* (later just: *equation morphism*) in an object A we mean a morphism $e : X \longrightarrow HX + A$ of $\mathcal{A}$, where X is a finitely presentable object of $\mathcal{A}$.

Suppose that A is an underlying object of an H-algebra $\alpha : HA \longrightarrow A$. Then by a *solution* of e in the algebra A is meant a morphism $e^\dagger : X \longrightarrow A$ in $\mathcal{A}$ such that the square

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha, A]} \\ HX + A & \xrightarrow[He^\dagger + A]{} & HA + A \end{array} \tag{2.1}$$

commutes.

An H-algebra is called *iterative* provided that every finitary flat equation morphism has a unique solution.

Example 2.5. The algebra $T_\Sigma Y$ of all Σ-*trees on* Y (i. e., trees with leaves labelled by constant symbols in Σ_0 or by elements of Y, and inner nodes with n children labelled in Σ_n) is iterative. And so is the subalgebra $R_\Sigma Y$ of all rational trees on Y.

Example 2.6. Groups, lattices etc. considered as Σ-algebras are seldom iterative. For example, if a group is iterative, then its unique element is the unit element 1, since the recursive equations $x \approx x \cdot y,\ y \approx 1$ have a unique solution. If a lattice is iterative, then it has a unique element: consider $x \approx x \vee x$.

Example 2.7. The algebra of addition on the set

$$\widetilde{\mathbb{N}} = \{\, 1, 2, 3, \dots \,\} \cup \{\, \infty \,\}$$

is iterative (and "almost classical"). (Observe that 0 is not included. This is forced by the uniqueness of solutions of $x \approx x + x$.)

To prove the iterativity of $\widetilde{\mathbb{N}}$ denote by $h : T_\Sigma \widetilde{\mathbb{N}} \longrightarrow \widetilde{\mathbb{N}}$ the homomorphism which to every finite tree assigns the result of computing the corresponding term in $\widetilde{\mathbb{N}}$ and to every infinite tree assigns ∞. Observe that the canonical embedding $\eta : \widetilde{\mathbb{N}} \longrightarrow T_\Sigma \widetilde{\mathbb{N}}$ satisfies $h \cdot \eta = id$. Let

$$e : X \longrightarrow X \times X + \widetilde{\mathbb{N}}$$

be an equation morphism. The derived equation morphism

$$\overline{e} \equiv X \xrightarrow{\ e\ } X \times X + \widetilde{\mathbb{N}} \xrightarrow{\ X \times X + \eta\ } X \times X + T_\Sigma \widetilde{\mathbb{N}}$$

has a unique solution $\overline{e}^\dagger : X \longrightarrow T_\Sigma \widetilde{\mathbb{N}}$ in the tree algebra. This yields a solution $e^\dagger$ in $\widetilde{\mathbb{N}}$ as follows:

$$e^\dagger \equiv X \xrightarrow{\ \overline{e}^\dagger\ } T_\Sigma \widetilde{\mathbb{N}} \xrightarrow{\ h\ } \widetilde{\mathbb{N}}\,.$$

To prove that solutions in $\widetilde{\mathbb{N}}$ are unique, let $e^\ddagger : X \longrightarrow \widetilde{\mathbb{N}}$ be a solution of e. For every $x \in X$ with $\overline{e}^\dagger(x)$ finite we have $e^\ddagger(x)$ as the computation of $\overline{e}^\dagger(x)$, i. e., $e^\ddagger(x) = e^\dagger(x)$ (easy proof by induction on the cardinality of the set of nodes of $\overline{e}^\dagger(x)$). And for every x with $\overline{e}^\dagger(x)$ infinite we prove $e^\ddagger(x) = \infty\ (= e^\dagger(x))$. This follows from the next Lemma since $\overline{e}^\dagger(x)$ has either infinitely many leaves or a complete binary subtree.

Lemma.

(1) Suppose that the tree $\overline{e}^\dagger(x)$ has (at least) k leaves labelled by $r_1, \dots, r_k \in \widetilde{\mathbb{N}}$, then

$$e^\dagger(x) \geq r_1 + \cdots + r_k\,.$$

(2) Suppose that the tree $\overline{e}^\dagger(x)$ has a node whose subtree is a complete binary tree (no leaves), then $e^\dagger(x) = \infty$.

Proof. (a) is proved by induction on the maximum depth d of the k leaves: The case $d = 0$, i. e., where $\overline{e}^\dagger(x)$ is a single root labelled by r_1, is clear: $e^\dagger(x) = r_1$. In the induction step let $d > 0$. Then certainly $e(x) \in X \times X$ say, $e(x) = (y_1, y_2)$, and each of the k leaves is a leaf of $\overline{e}^\dagger(y_i)$, $i = 1$ or 2. Since the maximum depth in $\overline{e}^\dagger(y_i)$ is one less than that in $\overline{e}^\dagger(x)$, we can use the induction hypothesis to conclude

$$e^\dagger(y_1) + e^\dagger(y_2) \geq r_1 + \cdots + r_k\,.$$

And from $e(x) = (y_1, y_2)$ we obtain, due to $e^\dagger = [\alpha, id] \cdot (H_\Sigma e^\dagger + id) \cdot e$,

$$e^\dagger(x) = e^\dagger(y_1) + e^\dagger(y_2) \geq r_1 + \cdots + r_k\,.$$

(b) is proved by induction on the depth of the given node j. The case $d = 0$ means that $\overline{e}^\dagger(x)$ is a complete binary tree, thus $e^\dagger(x)$ is an idempotent of $\widetilde{\mathbb{N}}$ — the unique idempotent is ∞. In the induction step we have $e(x) = (y, z)$ and the node j lies in $\overline{e}^\dagger(y)$ or $\overline{e}^\dagger(z)$ where it has smaller depth than in $\overline{e}^\dagger(x)$, thus $e^\dagger(y) = \infty$ or $e^\dagger(z) = \infty$. Consequently,

$$e^\dagger(x) = e^\dagger(y) + e^\dagger(z) = \infty\,.$$

□

Example 2.8. The algebra of addition of extended real numbers of the interval

$$I = (0, \infty]$$

is iterative.

The proof that equation morphisms have solutions is completely analogous to (2) above. The uniqueness is proved as follows: we first establish the above Lemma. Next we use (unlike in (2)!) the finiteness of the set X: since X is finite, the tree $\overline{e}^\dagger(x)$ is rational. If it has a subtree that is a complete binary tree, then $e^\dagger(x) = \infty$. Otherwise, every subtree of $\overline{e}^\dagger(x)$ contains a leaf, and the rationality of $\overline{e}^\dagger(x)$ then implies that infinitely many leaves of $\overline{e}^\dagger(x)$ carry the same label, say, $r \in I$. The Lemma, applied to k of these leaves, implies $e^\dagger(x) \geq k \cdot r$, for any $k = 1, 2, 3, \ldots$ — thus, $e^\dagger(x) = \infty$.

Remark 2.9. Uniqueness of solutions is sometimes subtle. In Example 2.7 above we need not assume that X is a finite set, but Example 2.8 would be false without this assumption: consider the system

$$\begin{aligned} x_0 &\approx x_1 + \frac{1}{2} \\ x_1 &\approx x_2 + \frac{1}{4} \\ x_2 &\approx x_3 + \frac{1}{8} \\ &\vdots \end{aligned}$$

One solution is $x_n^\dagger = \infty$ $(n \in \mathbb{N})$, another is $x_n^\dagger = 2^{-n}$ $(n \in \mathbb{N})$.

Example 2.10. *Unary algebras in* Set.

Let us consider the endofunctor

$$HA = \Sigma \times A$$

corresponding to unary Σ-algebras: every algebra $\alpha : \Sigma \times A \longrightarrow A$ is given by unary operations

$$s^A = \alpha(s, -) : A \longrightarrow A \quad \text{for } s \in \Sigma.$$

Such an algebra is iterative if and only if the operation

$$s_1^A \cdot s_2^A \cdot \cdots \cdot s_n^A : A \longrightarrow A$$

has a unique fixed point for every nonempty word $s_1 s_2 \cdots s_n$ over Σ.

In fact, the above condition is necessary because the solution of the following system

$$e : \{x_0, \ldots, x_{n-1}\} \longrightarrow \Sigma \times \{x_0, \ldots, x_{n-1}\} + A$$

where

$$e(x_i) = (s_i, x_{i+1}) \qquad \text{for } i < n-1, \text{ and } e(x_{n-1}) = (s_n, x_0)$$

is precisely a fixed point, a, of $s_1^A \cdot \cdots \cdot s_n^A$. More precisely, the corresponding map $e^\dagger : \{x_0, \ldots, x_{n-1}\} \longrightarrow A$ with

$$e^\dagger(x_i) = s_{i+1}^A \cdot \cdots \cdot s_n^A(a) \quad (i = 0, \ldots, n-1)$$

solves e.

To prove that the above condition is sufficient, consider a finitary equation morphism

$$e : X \longrightarrow \Sigma \times X\,.$$

Let us call a variable $x_0 \in X$ *cyclic* if the values of e always stay in the first summand, i.e., we have

$$e(x_i) = (s_{i+1}, x_{i+1}) \qquad i = 0, 1, 2, \ldots$$

for an infinite sequence $(s_n, x_n) \in \Sigma \times X$. Since X is finite, there exists $p < q$ with

$$x_p = x_q\,.$$

Every solution

$$e^\dagger : X \longrightarrow A$$

assigns to x_i elements $a_i = e^\dagger(x_i)$ such that

$$a_i = \alpha(s_{i+1}, a_{i+1})$$

in other words

$$a_i = s^A_{i+1}(a_{i+1})$$

Therefore $a_p = a_q$ implies that a_p is a fixed point of $s^A_{p+1} \cdot \cdots \cdot s^A_q$, and this fixed point determines the value

$$a_0 = s^A_1 \cdot \cdots \cdot s^A_p(a_p)\,.$$

Consequently, if the fixed point is unique, $e^\dagger(x_0)$ is uniquely determined.

The non-cyclic variables x_0 present no problem: here we have, for some $k \geq 0$,

$$\begin{aligned} e(x_i) &= (s_{i+1}, x_{i+1}) \qquad i = 0, \ldots, k-1 \\ e(x_k) &= a \in A \end{aligned}$$

which implies

$$e^\dagger(x_0) = s^A_1 \cdot \cdots \cdot s^A_k(a)\,.$$

Remark 2.11. In particular, for $Id : \mathsf{Set} \longrightarrow \mathsf{Set}$, an algebra $\alpha : A \longrightarrow A$ is iterative if and only if α has a unique fixed point and none of α^n, $n \geq 2$, has a different fixed point.

Example 2.12. *Ordered unary algebras.*

Here we consider, for a set Σ with discrete ordering, the endofunctor

$$HA = \Sigma \times A$$

on the category Pos of partially ordered sets and order-preserving functions. An ordered unary Σ-algebra is iterative if and only if the operation $s^A_1 \cdot \cdots \cdot s^A_n$ has a unique fixed point for every nonempty word $s_1 \cdots s_n$ over Σ.

The argument is as before, we just have to verify that the function

$$e^\dagger(x_0) = \begin{cases} s^A_1 \cdot \cdots \cdot s^A_p(a_p), & x_0 \text{ cyclic} \\ \\ s^A_1 \cdot \cdots \cdot s^A_p(a), & \text{else} \end{cases}$$

is order-preserving (whenever $e : X \longrightarrow \Sigma \times X + A$ is), which is easy.

Example 2.13. *Unary algebras in* Un.

Here the base category Un is that of unary algebras on one operation $\sigma_A : A \longrightarrow A$ and homomorphisms. We consider H-algebras for the identity endofunctor Id_{Un}. That is, algebras

$$\alpha : (A, \sigma_A) \longrightarrow (A, \sigma_A)\,,$$

where α is another unary operation on A, and since α is a homomorphism, it commutes with σ_A:

$$\alpha \cdot \sigma_A = \sigma_A \cdot \alpha\,.$$

Finitely presentable objects of Un are precisely the unary algebras given by finitely many generators and finitely many equations; for example, free algebras on n generators for $n \in \mathbb{N}$. We prove that an algebra is iterative if and only if

$$\sigma^k_A \alpha^n : A \longrightarrow A \text{ has a unique fixed point for all } n \geq 1 \text{ and } k \geq 0. \qquad (*)$$

The necessity of $(*)$ follows from solutions of the equation morphisms

$$e : X \longrightarrow X + A$$

where X is a free unary algebra on n generators, $x_1, \ldots, x_n$, and e is determined by

$$e(x_i) = x_{i+1} \quad \text{for } i < n, \quad e(x_n) = \sigma_X^k(x_1)$$

In fact, a solution $e^\dagger : X \longrightarrow A$ is determined by elements $a_i = e^\dagger(x_i)$, $i = 1, \ldots, n$ satisfying

$$a_i = \alpha(a_{i+1}) \quad \text{for } i < n,\ a_n = \sigma_A^k(a_1)$$

Thus, a_1 is a fixed point of $\sigma_A^k \alpha^n$, and conversely, every fixed point corresponds to a solution of e.

The sufficiency of $(*)$: given an equation morphism

$$e : X \longrightarrow X + A \quad \text{with } X \text{ generated by } y_1, \ldots, y_r$$

we can describe a solution analogously to in Example 2.10 above. Given a "non-cyclic" variable $x_0 \in X$, i.e., one with

$$\begin{aligned} e(x_i) &= x_{i+1} \qquad i = 0, \ldots, k-1 \\ e(x_k) &= a \in A \end{aligned}$$

we necessarily have $e^\dagger(x_k) = a$, $e^\dagger(x_{k-1}) = \alpha(a)$ etc., thus here

$$e^\dagger(x_0) = \alpha^k(a)$$

For a "cyclic" variable $x_0 \in X$ we have an infinite sequence $x_0, x_1, x_2, \ldots$ in X with $e(x_i) = x_{i+1}$. A solution $e^\dagger$ assigns to x_i an element $a_i \in A$ with

$$a_i = \alpha(a_{i+1}) = \alpha^2(a_{i+2}) = \ldots$$

On the other hand, we can express each x_i via the generators $y_1, \ldots, y_r$ in the form

$$x_i = \sigma_X^{c(i)}(y_{d(i)}) \quad c(i) \geq 0,\ d(i) \in \{\, 1, \ldots, r \,\}.$$

This implies $a_i = \sigma_A^{c(i)}(b_{d(i)})$ where $b_1, \ldots, b_r$ are the elements $e^\dagger(y_1), \ldots, e^\dagger(y_r)$. We can certainly choose $p < q$ such that

$$d(p) = d(q) \quad \text{and} \quad c(p) \leq c(q).$$

Then the equality $a_p = \alpha^{q-p}(a_q)$ yields

$$\sigma_A^{c(p)}(b_{d(p)}) = \alpha^{q-p}\sigma_A^{c(q)}(b_{d(p)})$$

Put $n = q - p$ and $k = c(q) - c(p)$ to conclude that

$$a_p = \sigma_A^{c(p)}(b_{d(p)}) \quad \text{is a fixed point of } \alpha^n \sigma_A^k.$$

Consequently, if a^* denotes the unique fixed point of $\alpha^n \sigma_A^k$, we conclude $a_1 = \alpha^p(a_p) = \alpha^p(a^*)$. Thus, we have to define

$$e^\dagger(x_0) = \alpha^p(a^*)$$

In summary, the unique solution of e is defined as follows:

$$e^\dagger(x_0) = \begin{cases} \alpha^k(a), & \text{if } x_0 \text{ is not cyclic} \\ \\ \alpha^p(a^*), & \text{if } x_0 \text{ is cyclic.} \end{cases}$$

Remark 2.14. We denote by

$$\mathsf{Alg}_{it}\ H$$

the category of all iterative algebras and all homomorphisms. The following lemma shows that this choice of morphisms is "right".

Lemma 2.15. (Homomorphisms = solutions-preserving morphisms.) *Let $h : A \longrightarrow B$ be an H-algebra homomorphism between iterative algebras. For every equation morphism $e : X \longrightarrow HX + A$ the solution of e in A yields a solution of the equation morphism*

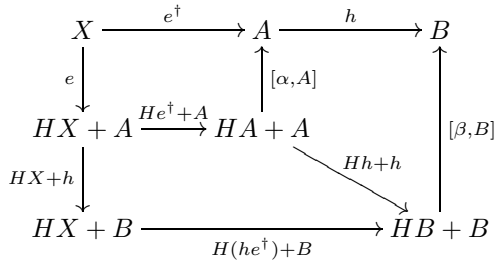
$$h \bullet e \equiv X \xrightarrow{\ e\ } HX + A \xrightarrow{\ HX + h\ } HX + B$$

in B via the commutative triangle

$$\begin{array}{ccc} & X & \\ {\scriptstyle e^\dagger}\swarrow & & \searrow{\scriptstyle (h \bullet e)^\dagger} \\ A & \xrightarrow[h]{} & B \end{array} \tag{2.2}$$

Conversely, any morphism h so that this triangle commutes for every equation morphism is an algebra homomorphism.

Proof. The following commutative diagram shows that $h \cdot e^\dagger$ solves $h \bullet e$:



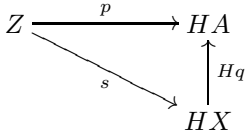
The upper left-hand part commutes since $e^\dagger$ is a solution of e, the right-hand part commutes since h is an H-algebra homomorphism, and the lower part is obvious. Thus, by the uniqueness of solutions we know that the triangle (2.2) commutes.

For the converse, let $\mathcal{A}_{fp}$ be a set of representative finitely presentable objects of $\mathcal{A}$, and let $\mathcal{A}_{fp}/A$ be the comma-category of all arrows $q : X \longrightarrow A$ with X in $\mathcal{A}_{fp}$. Since $\mathcal{A}$ is locally finitely presentable, A is a filtered colimit of the canonical diagram $D_A : \mathcal{A}_{fp}/A \longrightarrow \mathcal{A}$ given by $(q : X \longrightarrow A) \longmapsto X$.

Now $\mathcal{A}_{fp}$ is a generator of $\mathcal{A}$, thus, in order to prove the lemma it is sufficient to prove that for every morphism $p : Z \longrightarrow HA$ with Z in $\mathcal{A}_{fp}$ we have

$$h \cdot \alpha \cdot p = \beta \cdot Hh \cdot p. \tag{2.3}$$

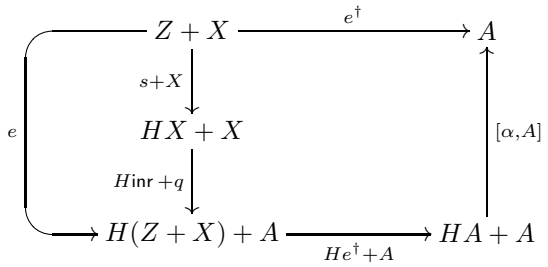
Since H is finitary, it preserves the above colimit D_A. This implies, since $\mathcal{A}(Z, -)$ preserves filtered colimits, that p has a factorization



for some $q : X \longrightarrow A$ in $\mathcal{A}_{fp}/A$ and some s. For the following equation morphism

$$e \equiv Z + X \xrightarrow{s+X} HX + X \xrightarrow{H\mathsf{inr}+q} H(Z + X) + A$$

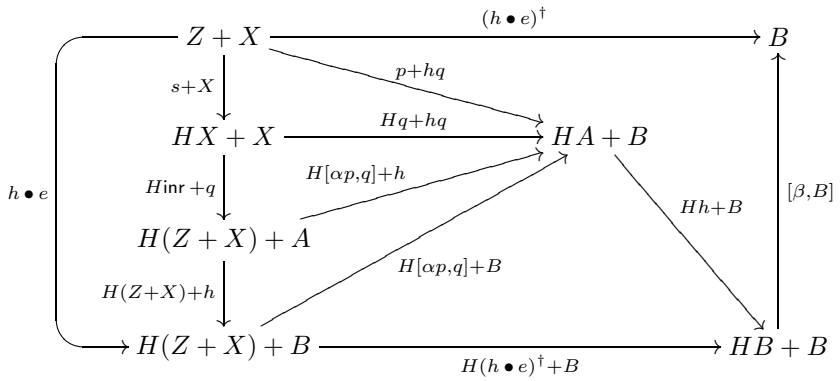
we have a commutative square



Consequently, $e^\dagger \cdot \mathsf{inr} = q$, and this implies $e^\dagger \cdot \mathsf{inl} = \alpha \cdot H(e^\dagger \cdot \mathsf{inr}) \cdot s = \alpha \cdot p$. By (2.2), we have $h \cdot e^\dagger = (h \bullet e)^\dagger$ and therefore

$$(h \bullet e)^\dagger = [h \cdot \alpha \cdot p, h \cdot q]. \tag{2.4}$$

On the other hand, consider the following diagram



It commutes: the outward square commutes since $(h \bullet e)^\dagger$ is a solution, for the lower triangle use equation (2.4), and the remaining triangles are trivial. Thus, the upper right-hand part commutes:

$$(h \bullet e)^\dagger = [\beta \cdot Hh \cdot p, h \cdot q]. \tag{2.5}$$

Now the left-hand components of (2.4) and (2.5) establish the desired equality (2.3). □

Proposition 2.16. *Iterative algebras are closed under limits and filtered colimits in* Alg H.

Proof. (1) Let (A, α) be a limit, in Alg H, of iterative algebras with a limit cone $h_i : (A, \alpha) \longrightarrow (A_i, \alpha_i)$, $i \in I$. It then easily follows that $A = \lim A_i$ in $\mathcal{A}$ with the limit cone $(h_i)_{i \in I}$. For every equation morphism $e : X \longrightarrow HX + A$ the uniqueness of its solution in A follows from Lemma 2.15: given $e^\dagger : X \longrightarrow A$, then each $h_i e^\dagger$ is the unique solution of $e_i = (HX + h_i) \cdot e$ in A_i, thus, $h_i e^\dagger$ is unique, and since $(h_i)_{i \in I}$ is a limit cone in $\mathcal{A}$, we conclude that $e^\dagger$ is unique. To prove the existence, let ${e_i}^\dagger : X \longrightarrow A_i$ denote the solution of e_i in A_i. This is a cone of the given diagram, i.e., for every connecting homomorphism

$$f : (A_i, \alpha_i) \longrightarrow (A_j, \alpha_j)$$

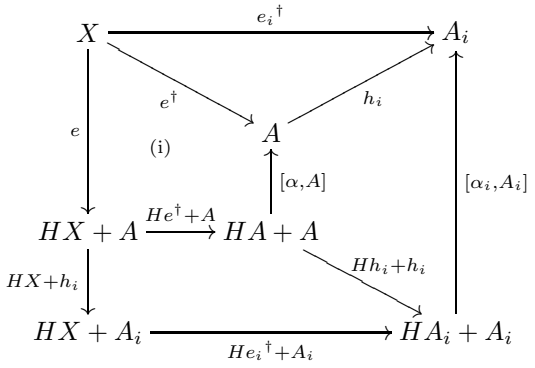
we have

$$f{e_i}^\dagger = {e_j}^\dagger.$$

This, again, follows from Lemma 2.15 and $fh_i = h_j$ (which implies $(HX + f) \cdot e_i = e_j$). Thus, there exists a unique morphism $e^\dagger : X \longrightarrow A$ with

$${e_i}^\dagger = h_i e^\dagger \quad (i \in I).$$

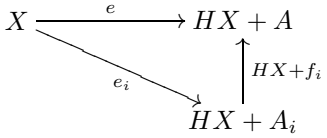
To prove that $e^\dagger$ solves e, it is sufficient to verify that $h_i e^\dagger = h_i \cdot [\alpha, A] \cdot (He^\dagger + A) \cdot e$ for all $i \in I$. In fact, the outer square of the following diagram



commutes, and so do the upper triangle, the right-hand and lower parts. Thus, part (i) commutes when extended by h_i as desired.

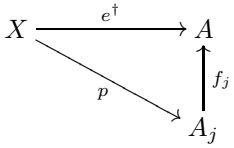
(2) Let (A, α) be a filtered colimit, in Alg H, of iterative algebras with a colimit cocone $f_i : (A_i, \alpha_i) \longrightarrow (A, \alpha)$, $i \in I$. Since H is finitary, filtered colimits of H-algebras are formed on the level of $\mathcal{A}$. Given an equation

morphism $e : X \longrightarrow HX + A = \mathrm{colim}(HX + A_i)$, since X is finitely presentable, e factors through one of the colimit morphisms $HX + f_i$:

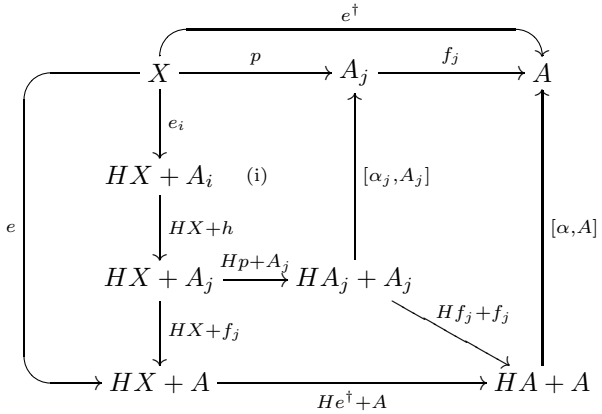


If ${e_i}^\dagger : X \longrightarrow A_i$ is the solution of e_i in A_i, then $f_i {e_i}^\dagger : X \longrightarrow A$ is a solution of e in A by Lemma 2.15.

Conversely, for every solution $e^\dagger : X \longrightarrow A$ of e in A we prove $e^\dagger = f_i {e_i}^\dagger$. Factorize $e^\dagger$ through one of the colimit morphisms:



Since the given diagram is filtered, we can suppose that the choice of $j \in I$ is such that a connecting homomorphism $h : (A_i, \alpha_i) \longrightarrow (A_j, \alpha_j)$ of our diagram exists. Then the morphism $e_j = (HX + h)e_i : X \longrightarrow HX + A_j$ has the solution ${e_j}^\dagger = p$. In fact, all parts of the following diagram



except (i) commute. Therefore (i) commutes when extended by f_j. By filteredness we can therefore suppose that (i) commutes (otherwise choose a connecting morphism $g : (A_j, \alpha_j) \longrightarrow (A_k, \alpha_k)$ equating the sides of (i) and work with k in lieu of j). But it follows from Lemma 2.15 that ${e_j}^\dagger = h{e_i}^\dagger$, therefore $p = h{e_i}^\dagger$. This proves

$$e^\dagger = f_j p = f_j h {e_i}^\dagger = f_i {e_i}^\dagger,$$

as desired. □

Corollary 2.17. *The category* $\mathsf{Alg}_{it}\ H$ *is a reflective subcategory of* $\mathsf{Alg}\ H$.

Proof. In fact, $\mathsf{Alg}\ H$ is locally finitely presentable, see Example 2.3(4). Thus, by the Reflection Theorem of [AR], every full subcategory closed under limits and filtered colimits is reflective. □

Corollary 2.18. *Every object of* $\mathcal{A}$ *generates a free iterative* H*-algebra.*

In other words, the natural forgetful functor $U : \mathsf{Alg}_{it}\ H \longrightarrow \mathcal{A}$ has a left adjoint.

Definition 2.19. The finitary monad on $\mathcal{A}$ formed by free iterative H-algebras is called the *rational monad* of H and is denoted by $\mathbb{R} = (R, \eta, \mu)$.

Thus, $\mathbb{R}$ is the monad of the above adjunction

$$\mathsf{Alg}_{it}\ H \underset{U}{\overset{R}{\underset{\longrightarrow}{\overset{\longleftarrow}{\bot}}}} \mathcal{A}$$

More detailed, for every object Z of $\mathcal{A}$ we denote by RZ a free iterative H-algebra on Z with the universal arrow

$$\eta_Z : Z \longrightarrow RZ\,,$$

and the algebra structure

$$\rho_Z : HRZ \longrightarrow RZ\,.$$

Then $\mu_Z : RRZ \longrightarrow RZ$ is the unique homomorphism of H-algebras with $\mu_Z \cdot \eta_{RZ} = id$.

Before turning to concrete examples of free iterative algebras, we will show that it is sufficient to describe the initial one:

Proposition 2.20. *For any object Y of $\mathcal{A}$ the following are equivalent:*

(1) *RY is an initial iterative algebra of $H(-) + Y$,*
(2) *RY is a free iterative H-algebra on Y.*

In fact, this was proved for completely iterative algebras in [Mi]; the proof for iterative algebras is the same.

Example 2.21. *The rational monad of Id.*

(a) For the identity functor on Set it follows from Example 2.10 that RZ is obtained from the free unary algebra $\mathbb{N} \times Z$ by adding a single element, say, a_0:

$$RZ = \mathbb{N} \times Z + 1$$

with

$$\eta_Z : Z \longrightarrow \mathbb{N} \times Z + 1 \quad z \longmapsto (0, z)$$

and

$$\rho_Z : \mathbb{N} \times Z + 1 \longrightarrow \mathbb{N} \times Z + 1 \quad (n, z) \longmapsto (n + 1, z),\ a_0 \longmapsto a_0.$$

(b) Analogously for the rational monad of Id on Pos we have

$$R(Z, \leq) = \mathbb{N} \times (Z, \leq) + 1 \quad \text{with } \mathbb{N} \text{ discretely ordered.}$$

This follows from Example 2.12.

(c) The rational monad of $Id : \mathsf{Un} \longrightarrow \mathsf{Un}$, see Example 2.13, is obtained as follows: given an object (Z, σ_Z) of Un, we first freely "add" a unary operation α which commutes with σ_Z by forming the algebra $Z \times \mathbb{N}$ with the operations σ given by $(z, n) \longmapsto (\sigma_Z(z), n)$ and α given by $(z, n) \longmapsto (z, n + 1)$. Then we add a single element, a_0, say, which is the joint fixed point of both operations. Thus,

$$R(Z, \sigma_Z) = (Z \times \mathbb{N} + 1, \sigma_{R(Z,\sigma_Z)})$$

where

$$\sigma_{R(Z,\sigma_Z)} : \begin{cases} (z, n) \longmapsto (\sigma_Z(z), n) \\ \\ a_0 \longmapsto a_0 \quad \text{where } 1 = \{a_0\}, \end{cases}$$

and with $\eta_{(Z,\sigma_Z)} : z \longmapsto (z, 0)$ and $\rho_{(Z,\sigma_Z)} : (z, n) \longmapsto (z, n + 1)$, $a_0 \longmapsto a_0$.

Example 2.22. *The rational monad of $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$.*

Recall from Example 2.5 that for every set Z the algebra $R_\Sigma Z$ of all rational Σ-trees over Z, i.e., Σ-trees over Z which have only finitely many subtrees (up to isomorphism), is iterative. As proved in [N], $R_\Sigma Z$ is a free iterative Σ-algebra on Z.

Corollary 2.23. *The rational monad $\mathbb{R}_\Sigma$ of the polynomial endofunctor H_Σ of Set is given by the formation of the Σ-algebras $R_\Sigma(Z)$ of all rational Σ-trees over Z.*

More precisely, the rational trees over Z (see Introduction) form an endofunctor $Z \longmapsto R_\Sigma(Z)$ of Set which is the underlying endofunctor of the monad $\mathbb{R}_\Sigma$. This follows from Proposition 2.20 and Example 2.22.

Example 2.24. The rational monad of $\mathcal{P}_{\mathrm{fin}} : \mathsf{Set} \longrightarrow \mathsf{Set}$, the finite power-set functor was described in [A_2]: it assigns to a set X the algebra of all rational strongly extensional finitely-branching trees (where "strongly extensional" means that every pair of distinct siblings define subtrees which are not bisimilar).

Remark 2.25. A special case of a recursive equation morphism is that where no parameters appear, i. e., simply coalgebras $e : X \longrightarrow HX$ with X finitely presentable. They appear in various contexts, e. g., in non-wellfounded set theory [BM] or, dually, in the theory of transitive sets [O]. Let us explain here why solutions of these special equation morphisms are not sufficient for our purposes. Let us (just in the present remark) call an algebra *weakly iterative* if every equation morphism $e : X \longrightarrow HX$, X finitely presentable, has a unique solution $e^\dagger : X \longrightarrow A$ (i. e., $e^\dagger = \alpha \cdot He^\dagger \cdot e$). For example in case $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$ represents a binary operation, $H_\Sigma X = X \times X$, the free iterative algebra $R_\Sigma\{\, a \,\}$ on one generator has the property that every equation $e : X \longrightarrow X \times X$ has the solution $e^\dagger : x \longmapsto t_0$, the constant function to the complete binary tree t_0. Consequently, every subalgebra of $R_\Sigma\{\, a \,\}$ containing t_0 and all finite trees is weakly iterative, although $R_\Sigma\{\, a \,\}$ has no proper iterative subalgebra containing finite trees.

3. A Coalgebraic Construction

The aim of this section is to describe an initial iterative H-algebra as a colimit of a simple diagram Eq in the given base category $\mathcal{A}$. We assume throughout this section that

(a) $\mathcal{A}$ is a locally finitely presentable category, see Definition 2.2,

and

(b) H is a finitary endofunctor of $\mathcal{A}$.

We choose a set $\mathcal{A}_{fp}$ of representatives of finitely presentable objects of $\mathcal{A}$ w.r.t. isomorphism.

Recall that (a) and (b) allow a simple description of the *initial H-algebra* as a colimit of the ω-chain

$$0 \xrightarrow{t} H0 \xrightarrow{Ht} HH0 \xrightarrow{HHt} \cdots$$

where t is the unique morphism from 0, an initial object of $\mathcal{A}$. More precisely, if $I = \operatorname{colim}_{n<\omega} H^n 0$ is this colimit, then the chain above defines a canonical morphism $i : I \longrightarrow HI$ — and one proves that i is invertible, yielding an initial H-algebra structure on I, see [A$_1$].

Analogously, the initial iterative algebra will be proved to be a colimit of the diagram

$$\mathrm{Eq} : \mathsf{EQ} \longrightarrow \mathcal{A}$$

whose objects are all H-coalgebras carried by finitely presentable objects of $\mathcal{A}$:

$$e : X \longrightarrow HX \quad \text{with } X \text{ in } \mathcal{A}_{fp},$$

with the usual coalgebra homomorphisms as morphisms, and with Eq the obvious forgetful functor $e \longmapsto X$.

A colimit

$$R_0 = \operatorname{colim} \mathrm{Eq}$$

of this diagram (with colimit morphisms $e^\sharp : X \longrightarrow R_0$ for all $e : X \longrightarrow HX$ in EQ) yields, again, a canonical morphism

$$i : R_0 \longrightarrow HR_0$$

Namely, i is the unique morphism such that every $e^\sharp$ becomes a coalgebra homomorphism, i.e., the squares

$$\begin{array}{ccc} X & \xrightarrow{\quad e \quad} & HX \\ {\scriptstyle e^\sharp}\downarrow & & \downarrow{\scriptstyle He^\sharp} \\ R_0 & \xrightarrow[\quad i \quad]{} & HR_0 \end{array} \tag{3.1}$$

commute. (In fact, the forgetful functor $\mathsf{Coalg}\, H \longrightarrow \mathcal{A}$ creates colimits.) The aim of the present section is to prove the following

Theorem 3.1. *R_0 is the initial iterative H-algebra. More precisely, the morphism i is an isomorphism and $i^{-1} : HR_0 \longrightarrow R_0$ is an initial iterative H-algebra.*

We establish some auxilliary facts first.

Remark 3.2. The diagram Eq is filtered. In fact, the category of all coalgebras is cocomplete, with colimits formed at the level of $\mathcal{A}$. Since $\mathcal{A}_{fp}$ is well-known to be closed under finite colimits, it follows that the category EQ is closed under finite colimits in the category of all H-coalgebras — thus, EQ is finitely cocomplete, hence, filtered.

Consequently, H preserves the colimit of Eq:

$$HR_0 = \operatorname{colim} H \cdot \mathrm{Eq}$$

with the colimit cocone $He^\sharp$.

Lemma 3.3. *$i : R_0 \longrightarrow HR_0$ is an isomorphism.*

Proof. (a) We define a morphism

$$j : HR_0 \longrightarrow R_0$$

We use the fact that in a locally finitely presentable category the given object HR_0 is a colimit of the diagram of all arrows $p : P \longrightarrow HR_0$ where P is in $\mathcal{A}_{fp}$. More precisely, let $\mathcal{A}_{fp}/HR_0$ denote the comma-category (of all these arrows p), then the forgetful functor $D_{HR_0} : \mathcal{A}_{fp}/HR_0 \longrightarrow \mathcal{A}$ has, in $\mathcal{A}$, the colimit cocone formed by all $p : P \longrightarrow HR_0$. Thus, in order to define j we need to define morphisms $jp : P \longrightarrow R_0$ forming a

cocone of the diagram D_{HR_0}. We know that HR_0 is a filtered colimit of $H \cdot \mathrm{Eq}$ and that $\mathcal{A}(P, -)$ preserves this colimit, since P is in $\mathcal{A}_{fp}$. Therefore, p factors through one of the colimit morphisms

$$\begin{array}{ccc} P & \xrightarrow{p} & HR_0 \\ & \searrow^{p'} & \uparrow{\scriptstyle Hg^\sharp} \\ & & HW \end{array} \tag{3.2}$$

for some $g : W \longrightarrow HW$ in EQ. We form a new object

$$e_{p'} \equiv P + W \xrightarrow{[p',g]} HW \xrightarrow{H\mathsf{inr}} H(P + W)$$

of EQ and define j to be the unique morphism such that the following square

$$\begin{array}{ccc} P & \xrightarrow{\mathsf{inl}} & P + W \\ {\scriptstyle p}\downarrow & & \downarrow{\scriptstyle {e_{p'}}^\sharp} \\ HR_0 & \xrightarrow[j]{} & R_0 \end{array} \tag{3.3}$$

commutes for every p in $\mathcal{A}_{fp}/HR_0$. To prove that j is well-defined we need to show that

(i) ${e_{p'}}^\sharp \cdot \mathsf{inl}$ is independent of the choice of factorization (3.2),

and

(ii) the morphisms ${e_{p'}}^\sharp \cdot \mathsf{inl}$ form a cocone of $\mathcal{A}_{fp}/HR_0$.

For (i), consider another factorization

$$\begin{array}{ccc} P & \xrightarrow{p} & HR_0 \\ & \searrow^{q'} & \uparrow{\scriptstyle Hf^\sharp} \\ & & HV \end{array}$$

for $f : V \longrightarrow HV$ in EQ. Using the fact that the diagram $H\mathrm{Eq}$ is filtered, we conclude that, without loss of generality, this new factorization can be assumed to posses a coalgebra homomorphism $h : W \longrightarrow V$ from the first one with $q' = Hh \cdot p'$:

$$\begin{array}{ccccc} W & \xrightarrow{g} & HW & \overset{p'}{\nwarrow} & \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh} & & P \\ V & \xrightarrow[f]{} & HV & \underset{q'}{\swarrow} & \end{array}$$

This yields a coalgebra homomorphism $P + h$ from $e_{p'}$ to $e_{q'}$:

$$\begin{array}{ccccc} P + W & \xrightarrow{[p',g]} & HW & \xrightarrow{H\mathsf{inr}} & H(P + W) \\ {\scriptstyle P+h}\downarrow & & \downarrow{\scriptstyle Hh} & & \downarrow{\scriptstyle H(P+h)} \\ P + V & \xrightarrow[[q',f]]{} & HV & \xrightarrow[H\mathsf{inr}]{} & H(P + V) \end{array}$$

which proves

$${e_{p'}}^\sharp = {e_{q'}}^\sharp \cdot (P + h)\,.$$

Consequently,

$${e_{p'}}^\sharp \cdot \mathsf{inl} = {e_{q'}}^\sharp \cdot (P + h) \cdot \mathsf{inl} = {e_{q'}}^\sharp \cdot \mathsf{inl}$$

as requested.

To prove (ii), consider a morphism r in $\mathcal{A}_{fp}/HR_0$:

$$\begin{array}{ccc} Q & \xrightarrow{r} & P \\ & {\scriptstyle q}\searrow \quad \swarrow{\scriptstyle p} & \\ & HR_0 & \end{array}$$

We have defined $jp = e_{p'}{}^\sharp \cdot \mathsf{inl}$ for the factorization (3.2) and, due to (i) above, we can use the factorization

$$q = Hg^\sharp \cdot (p' \cdot r)$$

for the definition of $jq = e_{p'r}{}^\sharp \cdot \mathsf{inl}$. It is our task to prove

$$e_{p'}{}^\sharp \cdot \mathsf{inl} \cdot r = e_{p'r}{}^\sharp \cdot \mathsf{inl}. \tag{3.4}$$

Observe that $r + W$ is a coalgebra homomorphism from $e_{p'r}$ to $e_{p'}$:

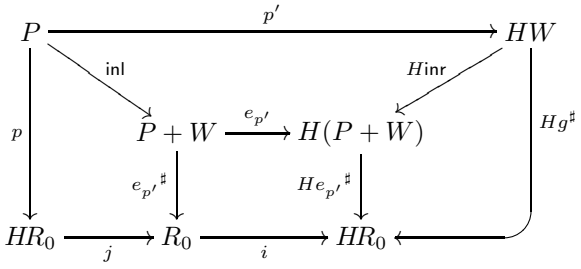
$$\begin{array}{ccccc} Q+W & \xrightarrow{[p'r,g]} & HW & \xrightarrow{H\mathsf{inr}} & H(Q+W) \\ {\scriptstyle r+W}\downarrow & & \| & & \downarrow{\scriptstyle H(r+W)} \\ P+W & \xrightarrow[[p',g]]{} & HW & \xrightarrow[H\mathsf{inr}]{} & H(P+W) \end{array}$$

Thus $e_{p'r}{}^\sharp = e_{p'}{}^\sharp \cdot (r + W)$ which proves (3.4).

(b) The proof of $ij = id$. It is our task to prove that $ijp = p$ for every $p : P \longrightarrow HR_0$ in $\mathcal{A}_{fp}/HR_0$. Observe that $\mathsf{inr} : W \longrightarrow P + W$ is a coalgebra homomorphism from $g : W \longrightarrow HW$ to $e_{p'} : P + W \longrightarrow H(P + W)$, thus,

$$g^\sharp = e_{p'}{}^\sharp \cdot \mathsf{inr}\ .$$

The desired equality $ijp = p$ follows from (3.2) and the fact that the following diagram



commutes.

(c) The proof of $ji = id$. It is our task to prove that $jie^\sharp = e^\sharp$ for every $e : X \longrightarrow HX$ in EQ. In order to do so, apply (3.3) to $p = He^\sharp \cdot e : X \longrightarrow HR_0$ with $p' = e$ and $g = e$ to obtain

$$j \cdot He^\sharp \cdot e = e_{p'}{}^\sharp \cdot \mathsf{inl} \tag{3.5}$$

for $e_{p'} \equiv X + X \xrightarrow{[e,e]} HX \xrightarrow{H\mathsf{inr}} H(X + X)$. It is easily checked that the codiagonal $\nabla : X + X \longrightarrow X$ is a coalgebra homomorphism from $e_{p'}$ to e, thus,

$$e^\sharp \cdot \nabla = e_{p'}{}^\sharp.$$

Now use $i \cdot e^\sharp = He^\sharp \cdot e$, see (3.1), and (3.5) to conclude

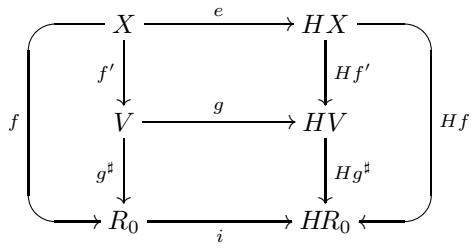
$$j \cdot (i \cdot e^\sharp) = j \cdot He^\sharp \cdot e = e_{p'}{}^\sharp \cdot \mathsf{inl} = e^\sharp \cdot \nabla \cdot \mathsf{inl} = e^\sharp.$$

□

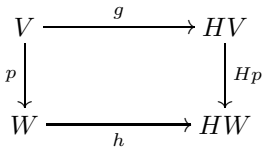
Remark 3.4. The coalgebra homomorphisms of (3.1) are unique: given an object $e : X \longrightarrow HX$ of EQ and a coalgebra homomorphism into R_0

$$\begin{array}{ccc} X & \xrightarrow{e} & HX \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle Hf} \\ R_0 & \xrightarrow[i]{} & HR_0 \end{array}$$

then $f = e^\sharp$. In fact, since X is finitely presentable, the morphism $f : X \longrightarrow \operatorname{colim} \mathrm{Eq}$ factors through the colimit morphism $g^\sharp$ for some $g : V \longrightarrow HV$: $f = g^\sharp f'$. In the following diagram



the outward square commutes, and so do all inner parts except possibly for the upper square. This implies that $Hg^\sharp$ merges the two sides of that square. Now $Hg^\sharp$ is a colimit morphism of $HR_0 = H \operatorname{colim} \mathrm{Eq} = \operatorname{colim} H\mathrm{Eq}$ (recall that Eq is a filtered diagram; thus H preserves its colimit). Since X is finitely presentable, $\mathcal{A}(X, -)$ preserves the colimit of $H\mathrm{Eq}$ — thus, if $Hg^\sharp$ merges two morphisms, then one of the connecting maps Hp, where p is a morphism in EQ, i. e., the following square



commutes, also merges those morphisms. That is, we have

$$Hp \cdot (Hf' \cdot e) = Hp \cdot (g \cdot f'),$$

from which we conclude that pf' is a morphism of EQ from e to h since

$$H(p \cdot f') \cdot e = Hp \cdot g \cdot f' = h \cdot (p \cdot f')\,.$$

Thus, $e^\sharp = h^\sharp \cdot (pf')$. Now p being a morphism of EQ implies $g^\sharp = h^\sharp \cdot p$, and consequently

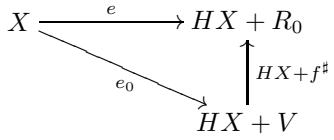
$$f = g^\sharp f' = h^\sharp p f' = e^\sharp.$$

Lemma 3.5. *The H-algebra $i^{-1} : HR_0 \longrightarrow R_0$ is iterative.*

Proof. (1) *Existence of solutions.* For every equation morphism

$$e : X \longrightarrow HX + R_0 = \operatorname{colim}(HX + \mathrm{Eq})$$

there exists, since X is finitely presentable, a factorization through the colimit morphism $HX + f^\sharp$ (for some $f : V \longrightarrow HV$ in EQ):



(3.6)

Recall from 2.1 that $\mathsf{can} : HX + HV \longrightarrow H(X + V)$ denotes the canonical morphism. Define a new object, $\overline{e}$, of EQ as follows:

$$\overline{e} \equiv X + V \xrightarrow{[e_0, \mathsf{inr}]} HX + V \xrightarrow{HX+f} HX + HV \xrightarrow{\mathsf{can}} H(X + V)\,. \tag{3.7}$$

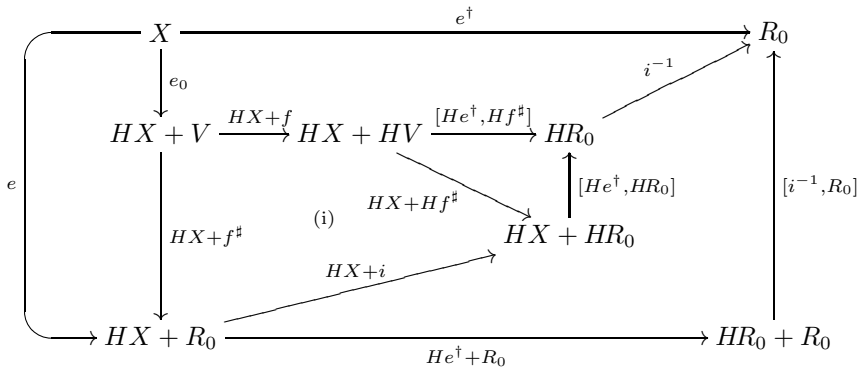
Observe that

$$f^\sharp = \overline{e}^\sharp \cdot \mathsf{inr} \tag{3.8}$$

because $\mathsf{inr} : V \longrightarrow X + V$ is a coalgebra morphism (in EQ) from f to $\overline{e}$. We define a solution of e by

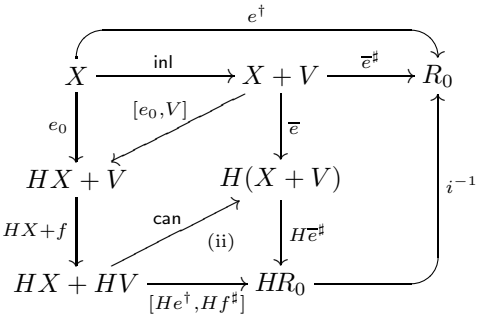
$$e^\dagger \equiv X \xrightarrow{\mathsf{inl}} X + V \xrightarrow{\overline{e}^\sharp} R_0\,. \tag{3.9}$$

In fact, in the following diagram



(3.10)

all inner parts commute: see (3.6) for the left-hand part, (3.1) for part (i), whereas the right-hand part commutes trivially (analyze the two components separately) and so does the middle triangle. It remains to verify the upper part: here we use (3.1) and (3.7) to conclude that the following diagram

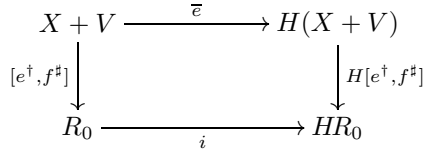


commutes. In fact, the left-hand component of (ii) commutes by definition of $e^\dagger$ and the right-hand one does by (3.8). Thus, (3.10) commutes, proving that $e^\dagger$ is a solution of e.

(2) *Uniqueness.* Suppose that $e^\dagger : X \longrightarrow R_0$ is a solution of e. Then in (3.10) the outward square commutes. Since all the inner parts except the upper one commute, this proves that the upper part commutes, too. Consequently,

$$i \cdot e^\dagger = [He^\dagger, Hf^\sharp] \cdot (HX + f) \cdot e_0 = H[e^\dagger, f^\sharp] \cdot \overline{e} \cdot \mathsf{inl}\ .$$

This equality implies that in the following square

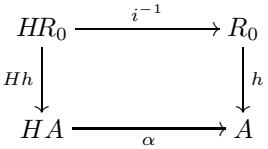


the left-hand components commute. Since $\overline{e} \cdot \mathsf{inr} = H\mathsf{inr} \cdot f$, the right-hand ones commute by (3.1). Therefore, the square commutes, which, by Remark 3.4, proves

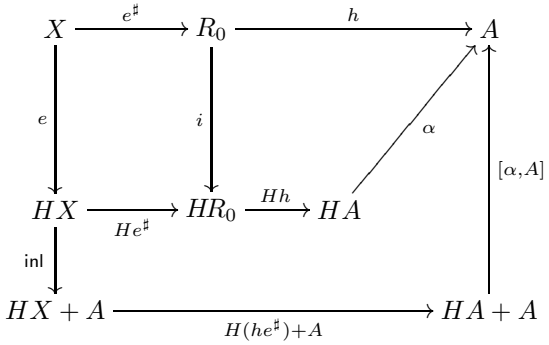
$$\overline{e}^\sharp = [e^\dagger, f^\sharp]\,.$$

Thus, the given solution is the previous one: $e^\dagger = \overline{e}^\sharp \cdot \mathsf{inl}\,.$ □

Proof of Theorem 3.1. Let $\alpha : HA \longrightarrow A$ be an iterative H-algebra. We prove first that there is at most one H-algebra homomorphism from R_0. Let



be a homomorphism. For every object $e : X \longrightarrow HX$ of EQ the following diagram



(3.11)

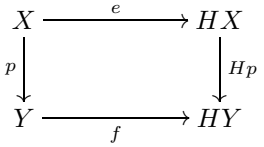
commutes, see (3.1), which proves that $he^\sharp$ is a solution of $\mathsf{inl}\, e$ in A.

This determines h uniquely, since the $e^\sharp$'s form a colimit cocone of $R_0 = \operatorname{colim} \mathrm{Eq}$.

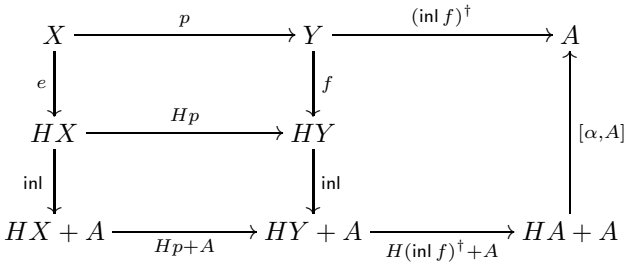
Conversely, let us define a morphism $h : R_0 \longrightarrow A$ by the above rule

$$he^\sharp = (\mathsf{inl}\, e)^\dagger \quad \text{for all } e : X \longrightarrow HX \text{ in } \mathsf{EQ}$$

where $(-)^\dagger$ is the unique solution in A. This is well-defined since the morphisms $(\mathsf{inl}\, e)^\dagger$ form a cocone of the diagram Eq: in fact, let



be a morphism of EQ. We prove that $(\mathsf{inl}\, f)^\dagger p$ is a solution of $\mathsf{inl}\, e$ by considering the corresponding diagram:



This proves

$$(\mathsf{inl}\, e)^\dagger = (\mathsf{inl}\, f)^\dagger p\,.$$

The morphism h above is a homomorphism of algebras because the diagram (3.11) commutes: the outward square commutes by definition of h, the upper left-hand square by (3.1), and the lower part is obvious. This shows that the upper right-hand part commutes when precomposed with $e^\sharp$, e in EQ. Since the $e^\sharp$'s form a colimit cocone, it follows that h is a homomorphism. □

Corollary 3.6. *A free iterative H-algebra RZ is a colimit*

$$RZ = \operatorname{colim} \mathrm{Eq}_Z$$

of the diagram

$$\mathrm{Eq}_Z : \mathsf{EQ}_Z \longrightarrow \mathcal{A}$$

where EQ_Z consists of all equation morphisms $e : X \longrightarrow HX+Z$, $X \in \mathcal{A}_{fp}$, and all coalgebra homomorphisms w.r.t. $H(-) + Z$, and Eq_Z sends e to X.

In fact, this is a consequence of Proposition 2.20 and Theorem 3.1.

Remark 3.7. We denote, again, the colimit morphisms of Eq_Z by

$$e^\sharp : X \longrightarrow RZ$$

for all $e : X \longrightarrow HX + Z$ in EQ_Z. The appropriate isomorphism is denoted by

$$i_Z : RZ \longrightarrow HRZ + Z$$

It is characterized by the fact that the two coproduct injections of $HRZ + Z$ are (in the notation of Definition 2.19)

$$\mathsf{inl} = i_Z \rho_Z \quad \text{and} \quad \mathsf{inr} = i_Z \eta_Z$$

In other words, $i_Z = [\rho_Z, \eta_Z]^{-1}$.

4. An Alternative Definition of Iterativity

In the Introduction we considered non-flat systems (1.1) of recursive equations for Σ-algebras. And we argued that, due to the possibility of flattening such a system, we will just have to consider the flat equation morphism $e : X \longrightarrow H_\Sigma X + A$. We are going to make that statement precise by showing that in iterative algebras (in general, not only in Set) much more general systems of recursive equations than the flat ones are uniquely solvable. This implies that, for polynomial endofunctors of Set, our definition of iterative algebras coincides with that presented by Evelyn Nelson [N]. And as we explain in the next section, this also implies that the rational monad is iterative in the sense of Calvin Elgot [E].

Let us first remark that the condition stated for (1.1) in the Introduction, that no right-hand side be a single variable, is substantial: the equation $x \approx x$ has a unique solution only in the trivial terminal algebras. Systems satisfying the above condition are called *guarded*.

We first consider guarded systems where the right-hand sides live in the free H-algebra (i. e., they are finite trees in case $H = H_\Sigma$). Such systems are called *finitary*.

Remark 4.1. Since H is finitary, free H-algebras exist, see [A$_1$]. We denote for every object X in $\mathcal{A}$ a free algebra by $\varphi^0_X : HFX \longrightarrow FX$ with universal arrow $\eta^0_X : X \longrightarrow FX$. This defines a monad $\mathbb{F} = (F, \eta^0, \mu^0)$ where the component μ^0_X is the unique homomorphism $\mu^0_X : FFX \longrightarrow FX$ with $\mu^0_X \cdot \eta^0_{FX} = id$. It is easy to see that analogously to Proposition 2.20, FX is an initial algebra of $H(-) + X$; thus, by Lambek's Lemma [L]

$$FX = HFX + X\,. \tag{4.1}$$

More precisely, the morphism

$$j_X = [\varphi^0_X, \eta^0_X] : HFX + X \longrightarrow FX$$

is an isomorphism. For every H-algebra $\alpha : HA \longrightarrow A$ we have the unique homomorphism

$$\widehat{\alpha} : FA \longrightarrow A \qquad \text{with} \quad \widehat{\alpha} \cdot \eta_A = id$$

(which, in case of H_Σ, is the computation of (finite) terms over A in the Σ-algebra A). This allows us to define solutions of finitary equations morphisms in A as follows:

Definition 4.2.

(1) By a *finitary equation morphism* in an object A is meant a morphism

$$e : X \longrightarrow F(X + A), \quad X \text{ finitely presentable.}$$

(2) We call e *guarded* provided that it factors through the summand $HF(X + A) + A$ of $F(X + A) = HF(X + A) + X + A$ (see (4.1) above):

$$\begin{array}{ccc} X & \xrightarrow{\;e\;} & F(X + A) \\ & \searrow & \uparrow{\scriptstyle [\varphi^0, \eta^0 \cdot \mathsf{inr}]} \\ & & HF(X + A) + A \end{array}$$

(3) Suppose that A is an underlying object of an H-algebra $\alpha : HA \longrightarrow A$. Then by a *solution* of e in the algebra A is meant a morphism $e^\dagger : X \longrightarrow A$ in $\mathcal{A}$ such that the square

$$\begin{array}{ccc} X & \xrightarrow{\;e^\dagger\;} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \widehat{\alpha}} \\ F(X + A) & \xrightarrow[F[e^\dagger, A]]{} & FA \end{array} \tag{4.2}$$

commutes.

Remark 4.3. The square (4.2) in Definition 4.2 means, for polynomial functors, that the assignment $e^\dagger$ of variables $x \in X$ to elements of A has the following property: form the "substitution" mapping $[e^\dagger, A] : X + A \longrightarrow A$ (which interprets the variables as $e^\dagger$ does, and leaves elements of A unchanged). Extend it to the unique homomorphism

$$\widehat{\alpha} \cdot F[e^\dagger, A] : F(X + A) \longrightarrow A$$

of the free algebra. Then the (formal) equations $x \approx e(x)$ become actual identities in A after the substitution $x \longmapsto e^\dagger(x)$ is performed for all $x \in X$, and the right-hand sides are computed in A. This is precisely the definition of solution of (1.1) in the Introduction.

Theorem 4.4. *An H-algebra A is iterative if and only if every guarded finitary equation morphism in A has a unique solution.*

The proof of Theorem 4.4 follows from the next result, generalizing "finitary" to "rational".

Definition 4.5. By a *rational equation morphism* in an object A we mean a morphism

$$e : X \longrightarrow R(X + A), \qquad X \text{ finitely presentable},$$

and e is called *guarded* if it factors through the summand $HR(X + A) + A$ of $R(X+A) = HR(X+A)+X+A$ (see Remark 3.7):

$$\begin{array}{ccc} X & \xrightarrow{\ e\ } & R(X + A) \\ & \searrow & \uparrow{\scriptstyle [\rho, \eta \cdot \mathsf{inr}]} \\ & & HR(X + A) + A \end{array}$$

Suppose that A is an underlying object of an iterative H-algebra $\alpha : HA \longrightarrow A$. We denote (analogously to $\widehat{\alpha}$ above) by

$$\widetilde{\alpha} : RA \longrightarrow A$$

the unique homomorphism of H-algebras with $\widetilde{\alpha} \cdot \eta_A = id$. Then by a *solution* of e in the iterative algebra A is meant a morphism $e^\dagger : X \longrightarrow A$ in $\mathcal{A}$ such that the square

$$\begin{array}{ccc} X & \xrightarrow{\ e^\dagger\ } & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \widetilde{\alpha}} \\ R(X + A) & \xrightarrow[R[e^\dagger, A]]{} & RA \end{array}$$

commutes.

Theorem 4.6. *If A is an iterative H-algebra, then every guarded rational equation morphism e in A has a unique solution.*

Proof. Let $\alpha : HA \longrightarrow A$ be an iterative algebra. Given a guarded rational equation morphism

$$\begin{array}{ccc} X & \xrightarrow{\ e\ } & R(X + A) \\ & \searrow^{e_0} & \uparrow{\scriptstyle [\rho_{X+A}, \eta_{X+A} \cdot \mathsf{inr}]} \\ & & HR(X + A) + A \end{array}$$

we will prove that e has a unique solution $e^\dagger$.

(1) *Existence.* Recall from Corollary 3.6 that $R(X + A) = \operatorname{colim} \mathrm{Eq}_{X+A}$ with colimit cocone $g^\sharp : W \longrightarrow R(X + A)$ for all $g : W \longrightarrow HW + X + A$ in EQ_{X+A}. Since this colimit is filtered and H is finitary, we have a filtered colimit

$$HR(X + A) + A = \operatorname{colim} H\mathrm{Eq}_{X+A} + A$$

with the colimit cocone formed by all $Hg^\sharp + A$. Since X is a finitely presentable object, the morphism

$$e_0 : X \longrightarrow \operatorname{colim} H\mathrm{Eq}_{X+A} + A$$

factors through the colimit cocone:

$$\begin{array}{ccc} X & \xrightarrow{\ e_0\ } & HR(X + A) + A \\ & \searrow^{w} & \uparrow{\scriptstyle Hg^\sharp + A} \\ & & HW + A \end{array}$$

for some object $g : W \longrightarrow HW + X + A$ of EQ_{X+A} and some morphism w.

We define a finitary flat equation morphism as follows:

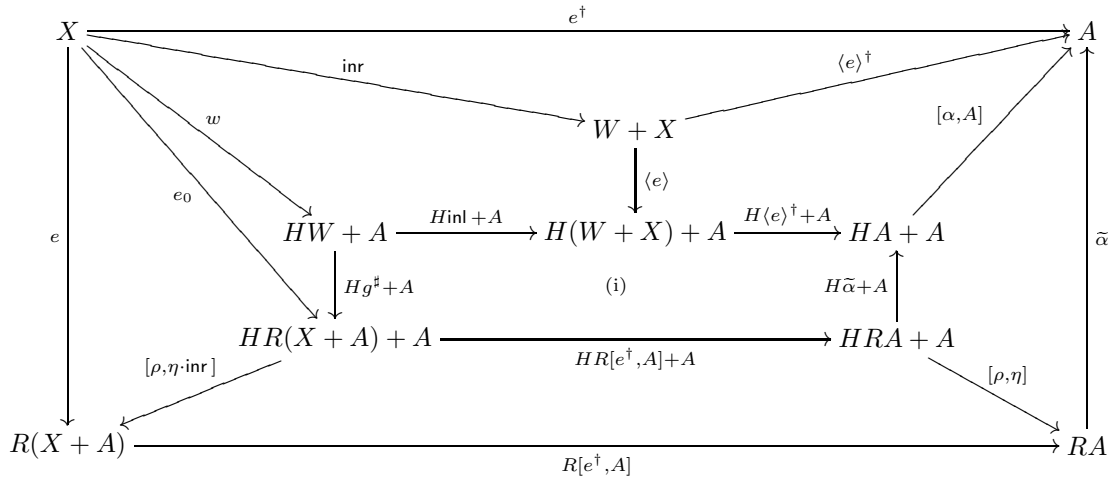
$$\langle e\rangle \equiv W + X \xrightarrow{[g,\mathsf{inm}]} HW + X + A \xrightarrow{[\mathsf{inl},w,\mathsf{inr}]} HW + A \xrightarrow{H\mathsf{inl}+A} H(W + X) + A \tag{4.3}$$

where $\mathsf{inm} : X \longrightarrow HW + X + A$ is the middle coproduct injection. We obtain a unique solution $\langle e\rangle^\dagger : W + X \longrightarrow A$ and prove that the following morphism

$$e^\dagger \equiv X \xrightarrow{\mathsf{inr}} W + X \xrightarrow{\langle e\rangle^\dagger} A \tag{4.4}$$

is a solution of e.

Indeed, consider the following diagram:



(4.5)

All of its parts, except the square (i), clearly commute. The right-hand component of (i) is obvious. To prove the commutativity of the left-hand component of (i), we remove H and show that the equation

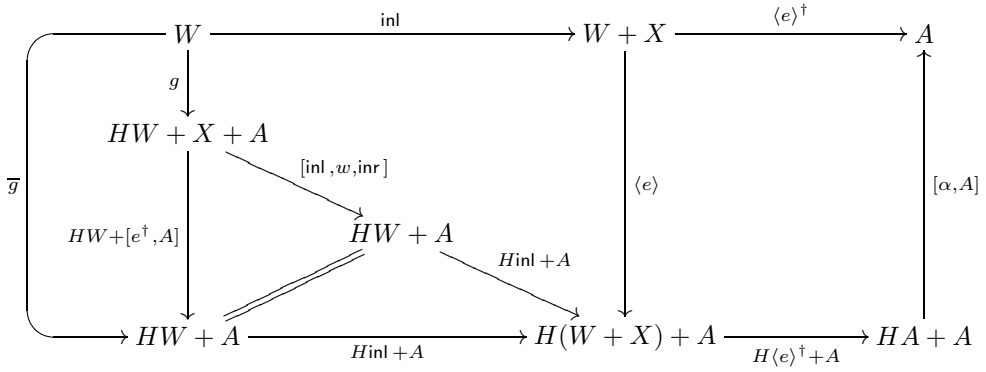
$$\langle e\rangle^\dagger \cdot \mathsf{inl} = \widetilde{\alpha} \cdot R[e^\dagger, A] \cdot g^\sharp \tag{4.6}$$

holds. To this end observe first that $\widetilde{\alpha} \cdot R[e^\dagger, A] : R(X + A) \longrightarrow A$ is an H-algebra homomorphism between iterative algebras extending $[e^\dagger, A]$. An inspection of the proof of Theorem 3.1 and Proposition 2.20 reveals that precomposing this homomorphism with the colimit injection $g^\sharp : W \longrightarrow R(X + A)$ yields the unique solution of the following equation morphism

$$\overline{g} \equiv W \xrightarrow{g} HW + X + A \xrightarrow{HW+[e^\dagger,A]} HW + A$$

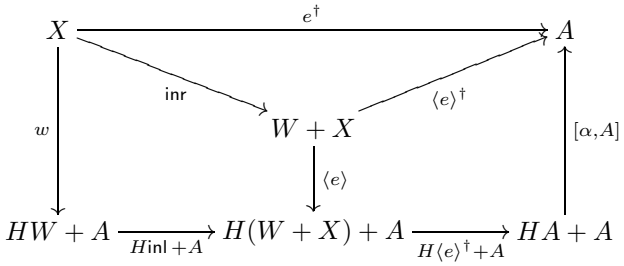
in the iterative algebra A.

Thus, to establish (4.6) it suffices to show that $\langle e\rangle^\dagger \cdot \mathsf{inl}$ is a solution of $\overline{g}$. In fact, the outward square of the following diagram



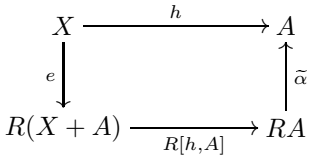
commutes. To prove this, observe that by (4.3) all parts except, perhaps, for the left-hand inner triangle, clearly commute. For that triangle consider the components of the coproduct separately. The left-hand and right-hand components are obviously commutative. We do not claim this for the middle component.

But this component commutes when extended to A in the upper right-hand corner. In fact, this yields the following square



which commutes: see the upper part of Diagram (4.5).

(2) *Uniqueness.* Let h be any solution of e, i.e., a morphism such that the following square



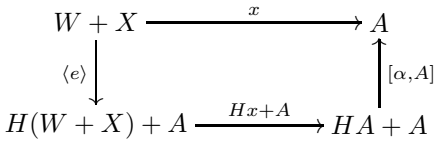
commutes. We shall show that

$$x \equiv W + X \xrightarrow{[\widetilde{\alpha}\cdot R[h,A]\cdot g^\sharp, h]} A$$

is a solution of $\langle e\rangle$ in A, therefore

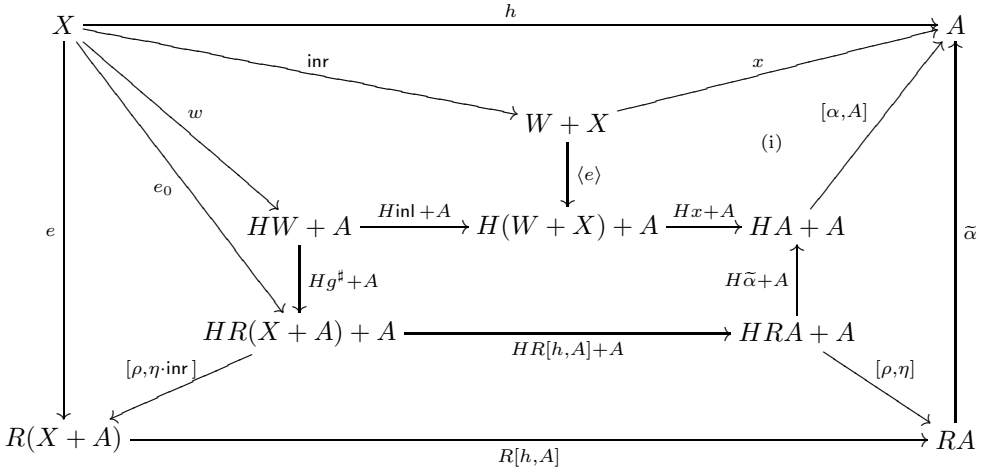
$$h = \langle e\rangle^\dagger \cdot \mathsf{inl} = e^\dagger$$

which completes the proof. Thus, it is our task to show that the following square



commutes.

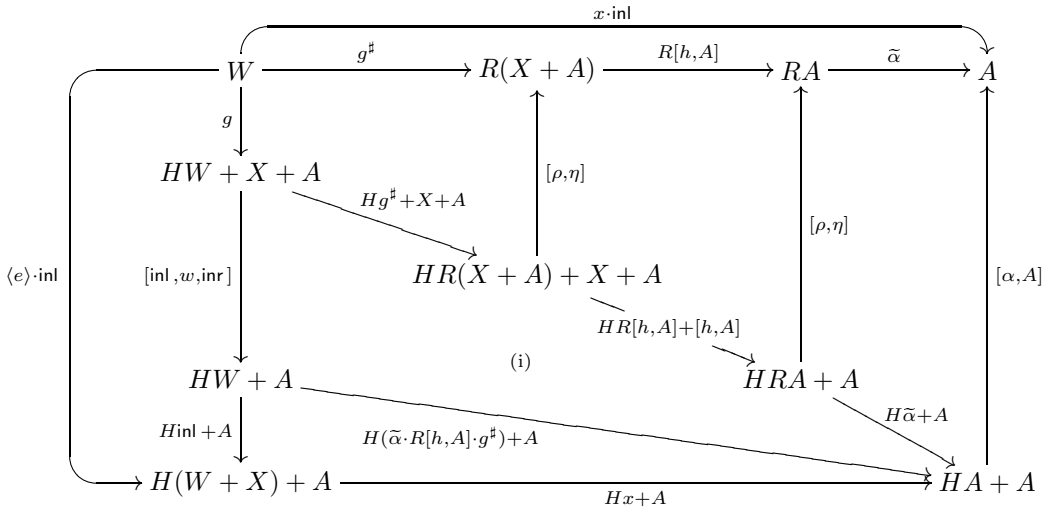
We consider the components of the coproduct $W + X$ separately. For the right-hand component we obtain the following commutative diagram



(4.7)

Since the outward square commutes, and all the inner parts but (i) clearly do, so must the right-hand component of part (i). (Notice that this diagram is precisely (4.5) with h for $e^\dagger$ and x for $\langle e\rangle^\dagger$.)

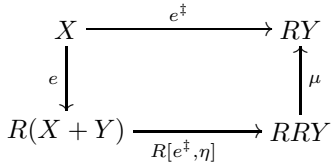
For the left-hand component consider the following diagram:



All of its parts commute, except possibly the middle component of (i), which commutes when extended by $[\alpha, A]$ to A in the upper right corner. In fact, this is easy to see by inspection of the upper three inner parts the of Diagram (4.7). □

The rational solution theorem we have proved in our previous work [AMV$_1$, AMV$_2$] is now an easy consequence of Theorem 4.6

Corollary 4.7. *Every rational guarded equation morphism $e : X \longrightarrow R(X + Y)$ has a unique solution in the algebra RY, i. e., there exists a unique $e^\ddagger : X \longrightarrow RY$ such that the square*

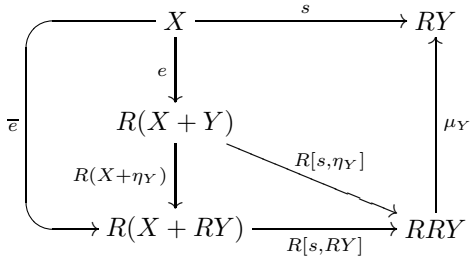


commutes.

Proof. Given a guarded rational equation morphism $e : X \longrightarrow R(X + Y)$ form the equation morphism

$$\overline{e} \equiv X \xrightarrow{e} R(X+Y) \xrightarrow{R(X+\eta_Y)} R(X+RY)\,.$$

This is a guarded equation morphism in the free iterative algebra RY. The result now follows from Theorem 4.6 applied to RY and to $\overline{e}$. In fact, there is a 1-1-correspondence between solutions of e and solutions of $\overline{e}$:



Observe first that $\widetilde{\rho_Y} = \mu_Y : RRY \longrightarrow RY$. Now since s is a solution of e, the upper inner part commutes, and equivalently, the outward square commutes, which is to say that s is a solution of $\overline{e}$. Since $\overline{e}$ has a unique solution, so does e. □

Proof of Theorem 4.4. (a) Sufficiency: let A be an iterative algebra. Denote by

$$\gamma : F \longrightarrow R$$

the natural transformation formed by the unique homomorphisms $\gamma_X : FX \longrightarrow RX$ of H-algebras with $\gamma_X \cdot \eta^0_X = \eta_X$. Observe that the following square

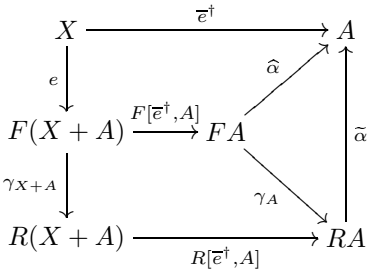
$$\begin{array}{ccc} FX & \xrightarrow{\gamma_X} & RX \\ {\scriptstyle [\varphi_X, \eta^0_X]}\uparrow & & \uparrow{\scriptstyle [\rho_X, \eta_X]} \\ HFX + X & \xrightarrow[H\gamma_X + X]{} & HRX + X \end{array} \tag{4.8}$$

commutes.

Given a guarded finitary equation morphism $e : X \longrightarrow F(X + A)$, we prove that a unique solution $e^\dagger$ exists. To this end form the rational equation morphism

$$\overline{e} \equiv X \xrightarrow{e} F(X + A) \xrightarrow{\gamma_{X+A}} R(X + A)$$

and observe that it is guarded (use (4.8)). The unique solution $\overline{e}^\dagger$ solves e. In fact, in the following diagram

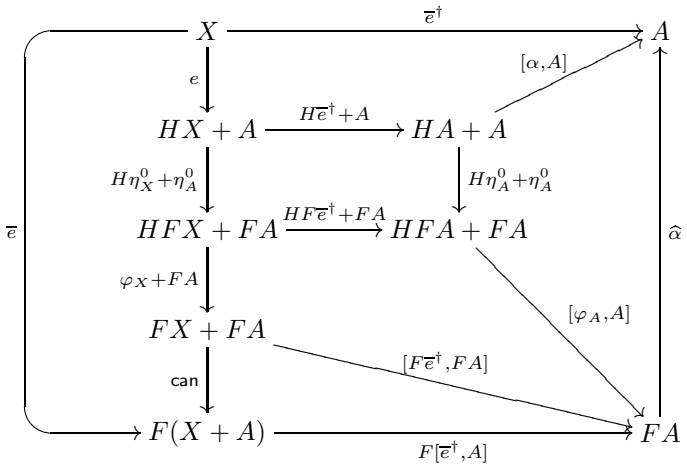


the outward square commutes (by definition of $\overline{e}^\dagger$), and the lower one does by the naturality of γ. The right-hand triangle commutes because both paths are homomorphisms extending id_A. Consequently, the upper square commutes, too. As for the uniqueness of solutions, suppose that in the above diagram $\overline{e}^\dagger : X \longrightarrow A$ denotes a solution of e. Then all inner parts of the diagram commute, thus, so does the outward square, whence $\overline{e}^\dagger$ is the unique solution of $\overline{e}$ (see Theorem 4.6).

(b) Necessity: if $\alpha : HA \longrightarrow A$ is an H-algebra such that every finitary equation morphism e has a unique solution, then A is iterative. In fact, given a flat equation morphism $e : X \longrightarrow HX + A$, denote by $\overline{e}$ the following finitary equation morphism

$$\overline{e} \equiv X \xrightarrow{e} HX + A \xrightarrow{H\eta^0_X + \eta^0_A} HFX + FA \xrightarrow{\varphi_X + FA} FX + FA \xrightarrow{\mathsf{can}} F(X + A)$$

It is easy to see that $\overline{e}$ is guarded. We obtain a unique solution $\overline{e}^\dagger : X \longrightarrow A$, and we prove that this solves e uniquely (in the sense of Definition 2.4). In other words, in the following diagram



the upper square commutes. In fact, the outward square commutes by definition of $\overline{e}^\dagger$, the right-hand one commutes because $\widehat{\alpha} \cdot \eta^0_A = id$ and (since $\widehat{\alpha}$ is a homomorphism)

$$\widehat{\alpha} \cdot \varphi_A \cdot H\eta^0_A = \alpha \cdot H\widehat{\alpha} \cdot H\eta^0_A = \alpha\,.$$

Since the remaining inner parts commute (by naturality of η and φ), the commutativity of the upper square follows.

To prove that e has a unique solution, suppose that in the above diagram $\overline{e}^\dagger : X \longrightarrow A$ denotes a solution of e. Then all inner parts of the diagram commute, thus, the outward square does. This shows that $\overline{e}^\dagger$ is the unique solution of $\overline{e}$. □

5. Free Iterative Monads

Assumptions 5.1. Throughout this section H denotes a finitary endofunctor of a locally finitely presentable category $\mathcal{A}$.

We are going to prove that the rational monad $\mathbb{R}$, introduced in Section 2, is iterative in the sense of Calvin Elgot, and that it can be characterized as a free iterative monad on H.

5.2. Iterative Monads. This is a concept that Elgot has introduced in [E] for the base category $\mathcal{A} = \mathsf{Set}$. He used the language of algebraic theories rather than monads, but we have proved in [AAMV] that the following concepts are equivalent to those of Elgot. For a monad $\mathbb{S} = (S, \eta, \mu)$ over Set we can form the complements of $\eta_X[X]$ in SX, say,

$$\sigma_X : S'X \longrightarrow SX$$

for all objects X. The monad $\mathbb{S}$ is called *ideal* provided $\sigma : S' \longrightarrow S$ is a subfunctor of S, and the monad multiplication has a domain-codomain restriction $\mu' : S'S \longrightarrow S'$. For general base categories in lieu of requiring a subfunctor S', we impose certain properties on μ' very similar to the monad laws for μ and η. The corresponding concept is as follows:

Definition 5.3. By an *ideal monad* is understood a sixtuple

$$\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$$

consisting of a monad (S, η, μ) and natural transformations $\sigma : S' \longrightarrow S$ and $\mu' : S'S \longrightarrow S'$ such that

(1) $S = S' + Id$ with coproduct injections σ and η, and
(2) The following three diagrams

$$\begin{array}{ccc} S' & \xrightarrow{S'\eta} & S'S \\ & \searrow^{=} & \downarrow{\scriptstyle \mu'} \\ & & S' \end{array} \qquad \begin{array}{ccc} S'SS & \xrightarrow{\mu' S} & S'S \\ {\scriptstyle S\mu}\downarrow & & \downarrow{\scriptstyle \mu'} \\ S'S & \xrightarrow[\mu']{} & S' \end{array} \qquad \begin{array}{ccc} S'S & \xrightarrow{\mu'} & S' \\ {\scriptstyle \sigma S}\downarrow & & \downarrow{\scriptstyle \sigma} \\ SS & \xrightarrow[\mu]{} & S \end{array} \tag{5.1}$$

commute.

Remark 5.4. Notice that the left-hand and middle diagrams in (5.1) express that the pair (S', μ') is a right $\mathbb{S}$-module, and the right-hand diagram states that σ is morphism of $\mathbb{S}$-modules from (S', μ') to (S, μ). The notion of a module appears for a monoidal category and a monoid in that category under the name action in [M],VII.4. We chose the name module to remind of the classical example of abelian groups; in this category, a monoid is precisely a ring R and an R-module is precisely a module of the ring R. Here we work in the monoidal category of endofunctors on $\mathcal{A}$ with composition as the tensor product and the identity functor as the tensor unit.

Examples 5.5.

(1) The rational monad is ideal. Recall from Remark 3.7 that $R = HR + Id$. Here we consider the natural transformation

$$\rho : HR \longrightarrow R$$

expressing the H-algebra structure $\rho_Z : HRZ \longrightarrow RZ$ of each RZ, see Definition 2.19. The "restriction" of μ here is simply

$$\mu' = H\mu : HRR \longrightarrow HR\,.$$

In fact, we know from Remark 3.7 that $RZ = HRZ + Z$ with the coproduct injections ρ_Z and η_Z. Next, $(HR, H\mu)$ is an $\mathbb{R}$-module: the first two diagram of (5.1) follow easily from the monad laws

for μ and η; furthermore, the third square

$$\begin{array}{ccc} HRR & \xrightarrow{H\mu} & HR \\ {\scriptstyle H\rho}\downarrow & & \downarrow{\scriptstyle \rho} \\ RR & \xrightarrow[\mu]{} & R \end{array}$$

commutes because each μ_Z is a homomorphism of H-algebras, see Definition 2.19.

(2) The free-algebra monad $\mathbb{F}$ of Section 4 is ideal. Here analogously, we use $F = HF + Id$, see (4.1).

(3) Classical algebraic theories (groups, lattices, etc.) are usually not ideal.

Definition 5.6. Let $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ be an ideal monad on $\mathcal{A}$.

(1) By a *finitary equation morphism* is meant a morphism

$$e : X \longrightarrow S(X + Y)$$

in $\mathcal{A}$ where X is a finitely presentable object ("of variables") and Y is any object ("of parameters").

(2) By a *solution* of e is meant a morphism

$$e^\dagger : X \longrightarrow SY$$

for which the square

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & SY \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu_Y} \\ S(X+Y) & \xrightarrow[S[e^\dagger, \eta_Y]]{} & SSY \end{array}$$

commutes.

(3) The equation morphism e is called *guarded* if it factors through the summand $S'(X + Y) + Y$ of $S(X + Y) = S'(X + Y) + X + Y$:

$$\begin{array}{ccc} X & \xrightarrow{e} & S(X+Y) \\ & \searrow & \uparrow{\scriptstyle [\sigma_{X+Y}, \eta_{X+Y}\,\mathsf{inr}]} \\ & & S'(X+Y)+Y \end{array}$$

(4) The ideal monad $\mathbb{S}$ is called *iterative* provided that every guarded finitary equation morphism has a unique solution.

Example 5.7. The rational monad of every finitary endofunctor is iterative, see Corollary 4.7.

Remark 5.8. Next we define morphisms of ideal monads. Whenever our base category $\mathcal{A}$ has the (very common) property that coproduct injections are monomorphic, then in an ideal monad $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ we automatically get a subfunctor $S' \hookrightarrow S$ and the module laws of (S', μ') follow automatically from the monad laws of S. This makes the definitions of morphisms easy and canonical: let $\mathbb{T} = (T, \eta^T, \mu^T, T', \tau, \mu'^T)$ be another ideal monad. An ideal monad morphism is a monad morphism

$$m : (S, \eta, \mu) \longrightarrow (T, \eta^T, \mu^T)$$

which has a restriction m' to the given subfunctors:

$$\begin{array}{ccc} S' & \xrightarrow{m'} & T' \\ {\scriptstyle \sigma}\downarrow & & \downarrow{\scriptstyle \tau} \\ S & \xrightarrow[m]{} & S' \end{array}$$

However, we do not want to impose any side conditions on $\mathcal{A}$. The prize is that ideal monad morphisms are defined as pairs (m, m'):

Definition 5.9.

(1) An *ideal monad morphism* from an ideal monad $(S, \eta, \mu, S', \sigma, \mu')$ to another one $(T, \eta^T, \mu^T, T', \tau, {\mu'}^T)$ is a pair (m, m') that consists of a monad morphism $m : (S, \eta, \mu) \longrightarrow (T, \eta^T, \mu^T)$ and a natural transformation $m' : S' \longrightarrow T'$ such that the diagrams

$$\begin{array}{ccc} S'S & \xrightarrow{m' * m} & T'T \\ {\scriptstyle \mu'}\downarrow & & \downarrow{\scriptstyle {\mu'}^T} \\ S' & \xrightarrow[m']{} & T' \end{array} \qquad \text{and} \qquad \begin{array}{ccc} S' & \xrightarrow{m'} & T' \\ {\scriptstyle \sigma}\downarrow & & \downarrow{\scriptstyle \tau} \\ S & \xrightarrow[m]{} & T \end{array} \tag{5.2}$$

commute.

(2) Given a functor H, a natural transformation $\lambda : H \longrightarrow S$ is called *ideal* provided that it factors through $\sigma : S' \longrightarrow S$: $\lambda = \sigma \cdot \lambda'$ for some transformation $\lambda' : H \longrightarrow S'$.

Remark 5.10. The left-hand square in Diagram (5.2) expresses that $m' : S' \longrightarrow T'$ is a module morphism with change of base m. The right-hand one together with the preservation of the unit $m \cdot \eta = \eta^T$ expresses that $m = m' + Id$. In fact, every ideal monad morphism is determined by its second component m'.

Example 5.11. For the rational monad $\mathbb{R}$, the natural transformation

$$\kappa \equiv H \xrightarrow{H\eta} HR \xrightarrow{\rho} R$$

is ideal.

Remark 5.12.

(1) We are going to prove that, for every finitary endofunctor H, the rational monad $\mathbb{R}$ is a free iterative monad on H. Since ideal monad morphisms are pairs, the freeness is expressed by a pair of equations. Notice, however, that under the assumption that coproduct injections in the base category $\mathcal{A}$ are monomorphic, see Remark 5.8, the freeness of $\mathbb{R}$ means what one expects: for every iterative monad $\mathbb{S}$ and every ideal natural transformation $\lambda : H \longrightarrow S$ there exists a unique ideal monad morphism $\overline{\lambda} : \mathbb{R} \longrightarrow \mathbb{S}$ such that $\overline{\lambda} \cdot \kappa = \lambda$. The formulation below refrains from the assumption that coproduct injections are monomorphic.

(2) Parts of the following proof are identical to the corresponding parts of Theorem 5.14 of [Mi]; we already mentioned that that paper was written parallel to ours. We decided to present a complete proof, without referencing to [Mi], for the convenience of the reader.

Theorem 5.13. (Rational Monad as a Free Iterative Monad.) *For every iterative monad $\mathbb{S}$ and every ideal natural transformation $\lambda : H \longrightarrow S$ there exists a unique ideal monad morphism $(\overline{\lambda}, \overline{\lambda}') : \mathbb{R} \longrightarrow \mathbb{S}$ such that the diagrams*

$$\begin{array}{ccc} H & \xrightarrow{H\eta} & HR \\ & {\scriptstyle \lambda'}\searrow & \downarrow{\scriptstyle \overline{\lambda}'} \\ & & S' \end{array} \qquad \textit{and} \qquad \begin{array}{ccc} H & \xrightarrow{\kappa} & R \\ & {\scriptstyle \lambda}\searrow & \downarrow{\scriptstyle \overline{\lambda}} \\ & & S \end{array} \tag{5.3}$$

commute.

Remark. Let us form the category $\mathsf{Fin}(\mathcal{A}, \mathcal{A})$ of all finitary endofunctors and natural transformations. For the category

$$\mathsf{FIM}(\mathcal{A})$$

of all finitary iterative monads (i.e., iterative monads $(S, \eta, \mu, S', \sigma, \mu')$ with S and S' finitary) and ideal monad morphisms we have a forgetful functor

$$U : \mathsf{FIM}(\mathcal{A}) \longrightarrow \mathsf{Fin}(\mathcal{A}, \mathcal{A}), \qquad \mathbb{S} \longmapsto S'$$

The above theorem states that U has a left adjoint, viz, the functor $H \longmapsto \mathbb{R}$.

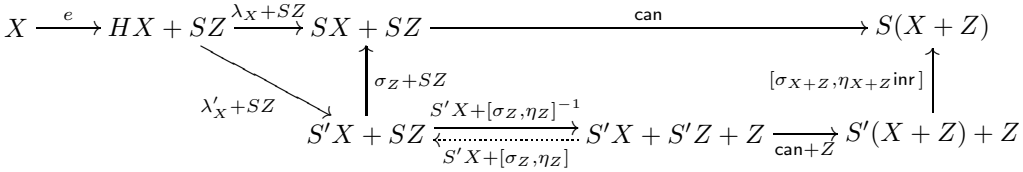
Proof. (1) For every object Z consider SZ as an H-algebra

$$HSZ \xrightarrow{\lambda_{SZ}} SSZ \xrightarrow{\mu_Z} SZ\,.$$

It is iterative. In fact, every equation morphism $e : X \longrightarrow HX + SZ$, X in $\mathcal{A}_{fp}$, yields the following equation morphism w.r.t. $\mathbb{S}$:

$$\overline{e} \equiv X \xrightarrow{\ e\ } HX + SZ \xrightarrow{\lambda_X + SZ} SX + SZ \xrightarrow{\ \mathsf{can}\ } S(X+Z)\,.$$

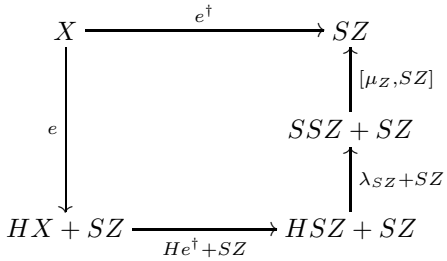
To verify that $\overline{e}$ is guarded, use the restriction $\lambda' : H \longrightarrow S'$ of λ:



To prove the commutativity of the square, consider the three components of $S'X + S'Z + Z$ separately, and use naturality of σ and η.

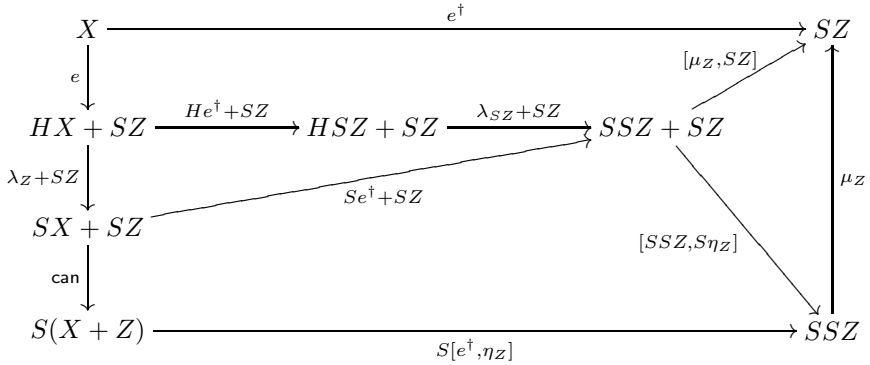
We prove that a morphism $e^\dagger : X \longrightarrow SZ$ is a solution of e in the H-algebra SZ if and only if it is a solution of $\overline{e}$ w.r.t. the iterative monad $\mathbb{S}$.

(1a) Let $e^\dagger$ be a solution of e in the algebra SZ, i.e., let



(5.4)

commute. We are to show that the following diagram



(5.5)

has the outward square commutative. The upper part is (5.4), the one directly below it is the naturality of λ. The lower part commutes obviously, and the right-hand one does due to $\mu_Z \cdot S\eta_Z = id$.

(1b) Let the outward square of (5.5) commute. Since the remaining three inner parts commute, so does the upper one, which is (5.4).

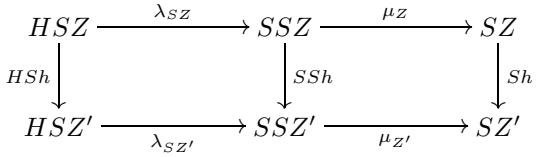
(2) *Existence of an ideal monad morphism $\overline{\lambda}$ such that (5.3) commute.* Denote by

$$\overline{\lambda}_Z : RZ \longrightarrow SZ$$

the unique homomorphism of H-algebras with

$$\overline{\lambda}_Z \cdot \eta_Z = \eta^S_Z\,.$$

We first observe that $\overline{\lambda}$ is a natural transformation. Given a morphism $h : Z \longrightarrow Z'$, then Sh is a homomorphism of H-algebras from SZ to SZ':

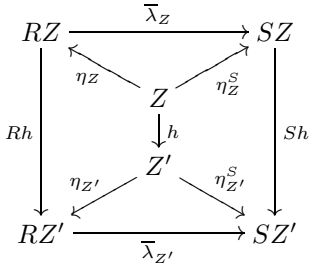


(5.6)

Thus, we have two parallel H-algebra homomorphisms from RZ to SZ':

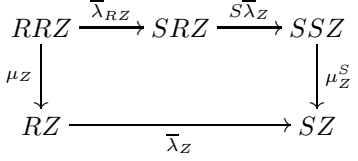
$$Sh \cdot \overline{\lambda}_Z \qquad \text{and} \qquad \overline{\lambda}_{Z'} \cdot Rh\,.$$

They agree when precomposed with η_Z:



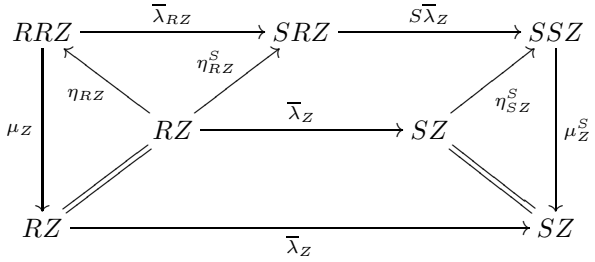
By the universal property of η_Z, and since SZ' is an iterative H-algebra, this proves that the above naturality square commutes.

Let us prove that $\overline{\lambda}$ is a monad morphism. Since $\overline{\lambda}\eta = \eta^S$ by definition, it remains to prove the commutativity of the following diagram



(5.7)

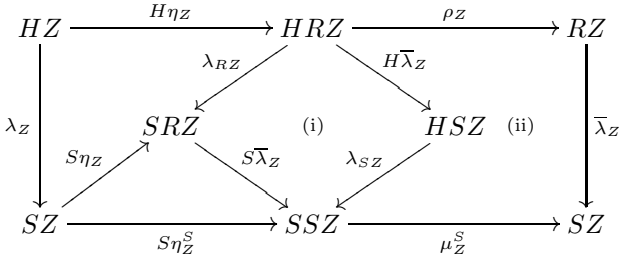
By (5.6), applied to $h = \overline{\lambda}_Z$, we see that $S\lambda_Z$ is a homomorphism of H-algebras. By the universal property of η_{RZ} it is sufficient to prove that (5.7) commutes when precomposed with η_{RZ}:



The equation

$$\lambda = \overline{\lambda} \cdot \kappa = \overline{\lambda} \cdot \rho \cdot H\eta$$

follows from the commutativity of the following diagram



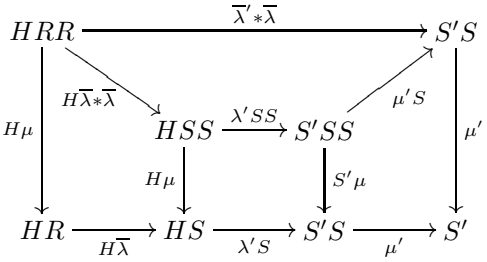
(5.8)

where (i) is naturality of λ and (ii) is clear since $\overline{\lambda}$ is a homomorphism. Now use that $\mu^S_Z \cdot S\eta^S_Z = id$.

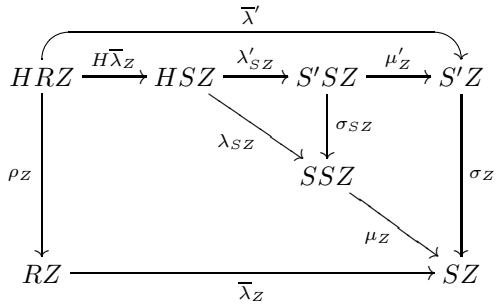
Thus, we have found a monad morphism $\overline{\lambda} : \mathbb{R} \longrightarrow \mathbb{S}$ with $\overline{\lambda} \cdot \kappa = \lambda$. It remains to verify that $\overline{\lambda}$ is part of an ideal monad morphism. Put

$$\overline{\lambda}' \equiv HR \xrightarrow{H\overline{\lambda}} HS \xrightarrow{\lambda' S} S'S \xrightarrow{\mu'} S'\,. \tag{5.9}$$

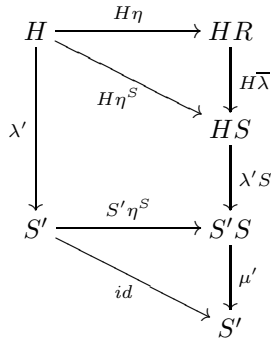
To see that the pair $(\overline{\lambda}, \overline{\lambda}')$ is an ideal monad morphism we have to verify the commutativity of the diagrams (5.2) for this pair. For the left-hand diagram of (5.2) consider the following diagram



The upper part clearly commutes by the definiton (5.9) of $\overline{\lambda}'$ and by the naturality of parallel composition. The other parts of the diagram are clear by invoking — from left to right — that $\overline{\lambda}$ is a monad morphism, the naturality of λ' and the module laws of S'. To verify the right-hand square of (5.2) consider the diagram



Finally, let us check the left-hand triangle of (5.3), i. e., we show that $\overline{\lambda}' \cdot H\eta = \lambda'$. To see this, consider the diagram



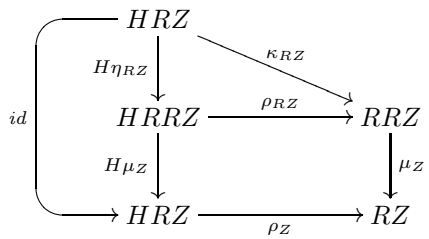
It commutes: for the upper triangle use that $\overline{\lambda}$ is a monad morphism, the middle part is naturality, and the lower triangle is the unit law of the $\mathbb{S}$-module S'.

(3) *Uniqueness of* $\overline{\lambda}$. Suppose that (m, m') is an ideal monad morphism from $\mathbb{R}$ to $\mathbb{S}$ such that the Diagrams (5.3) commute with (m, m') in lieu of $(\overline{\lambda}, \overline{\lambda}')$. We are going to show that for any object Z, m_Z is an H-algebra homomorphism extending η^S_Z, and then invoke the freeness of RZ as an iterative H-algebra, which implies that $m = \overline{\lambda}$, and then leads to the conclusion that $m' = \overline{\lambda}'$.

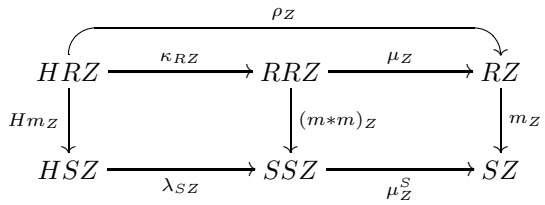
Firstly, notice that for any object Z we have

$$\rho_Z = \mu_Z \cdot \kappa_{RZ}\,. \tag{5.10}$$

Indeed, the diagram



commutes. Consequently, the following diagram

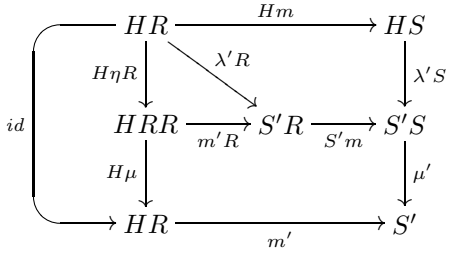


commutes: the upper part is (5.10), the right-hand square commutes since m is a monad morphism, and the left-hand one does since $m \cdot \kappa = \lambda$ and by naturality.

Thus $m_Z : RZ \longrightarrow SZ$ is an H-algebra homomorphism between iterative H-algebras such that $m_Z \cdot \eta_Z = \eta^S_Z$. This implies that $m = \overline{\lambda}$ and from this it follows that

$$m' = \mu' \cdot \lambda' S \cdot Hm = \mu' \cdot \lambda' S \cdot H\overline{\lambda}$$

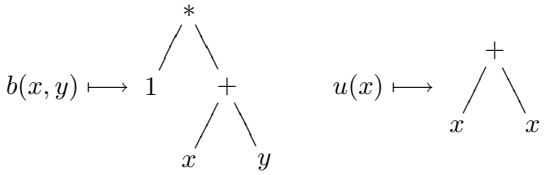
where the first equation holds due to the following diagram



The lower square commutes since m' is a module homomorphism with change of base m, the left-hand part by the unit law of the monad $\mathbb{R}$, the upper triangle by (5.3) and the upper right-hand part by naturality. This completes the proof. □

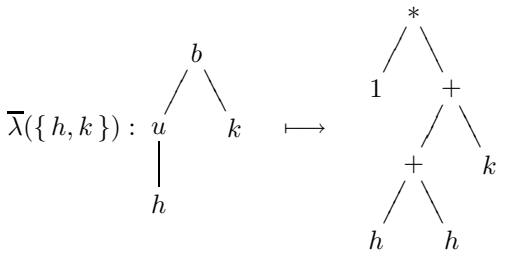
Remark 5.14. For polynomial endofunctors on Set, the freeness of $\mathbb{R}$ specializes to *second order substitution*, see [C], i. e., substitution of rational trees for operation symbols.

For example, consider a signature Σ with a binary operation symbol b, and a unary one u, and another signature Γ with two binary operation symbols $+$ and $*$ and a constant symbol 1. The assignment



(5.11)

of operation symbols in Σ to rational trees over Γ gives rise to a natural transformation $\lambda : H_\Sigma \longrightarrow R_\Gamma$. The induced monad morphism $\overline{\lambda} : \mathbb{R}_\Sigma \longrightarrow \mathbb{R}_\Gamma$ replaces, for any set of variables X, the operation symbols in trees of $R_\Sigma X$ according to λ. Example:



The requirement that λ be an ideal transformation means that no operation symbol of Σ is replaced by a single variable, i. e., that λ is a so-called *non-erasing* substitution.

Remark 5.15. We have defined a rational monad for every finitary endofunctor of a locally finitely presentable category. One may ask what happens if we "raise the index of presentability" to an uncountable regular cardinal λ. That is, what is the "λ-rational" monad of a λ-accessible endofunctor H (i.e., one, preserving λ-filtered colimits)?

It is easy to see that the main results above remain true if we systematically replace "finitely presentable" by "λ-presentable" and "finitary" by "λ-accessible". An H-algebra A might be called λ-*iterative* if every equation morphism $e : X \longrightarrow HX + A$ with X λ-presentable has a unique solution $e^\dagger : X \longrightarrow A$.

Then, for a λ-accessible endofunctor $H : \mathcal{A} \longrightarrow \mathcal{A}$, one can prove the following

(1) The category of all λ-iterative H-algebras is reflective in $\mathsf{Alg}\ H$.
(2) The resulting λ-accessible monad $\mathbb{R}^\lambda$ on $\mathcal{A}$ is a free λ-iterative monad on H. Again, λ-*iterative* means unique solvability of equations $X \longrightarrow R^\lambda(X+Z)$ with X λ-presentable. Moreover, $R^\lambda Z = \operatorname{colim} \mathrm{Eq}^\lambda_Z$, where Eq^λ_Z is the obvious modification of the diagram Eq_Z from Corollary 3.6.

However, in case of uncountable λ such a monad $\mathbb{R}$ coincides with the *completely iterative monad* $\mathbb{T}$ of H which has been described in [AAMV, Mi]. This monad $\mathbb{T}$ is given object-wise by final coalgebras of the endofunctor $H(-) + Z : \mathcal{A} \longrightarrow \mathcal{A}$. To show that $\mathbb{T} \cong \mathbb{R}^\lambda$, it therefore suffices to prove the following:

Proposition. *For uncountable λ, the object $R^\lambda Z$ is a final coalgebra of $H(-) + Z$. More precisely, the isomorphism $i_Z : R^\lambda Z \longrightarrow HR^\lambda Z + Z$ is a final coalgebra of $H(-) + Z$.*

Proof. We use the fact proven in [AP] that the category EQ^λ_Z is a dense full subcategory of the locally λ-presentable category of all coalgebras of $H(-) + Z$. Here we use uncountability of λ. Thus, it suffices to prove that for every

$$e : X \longrightarrow HX + Z$$

in EQ^λ_Z there exists a unique homomorphism into $i_Z : R^\lambda Z \longrightarrow HR^\lambda Z + Z$. Since the colimit injection $e^\sharp : X \longrightarrow R^\lambda Z$ is such a homomorphism, it remains to verify uniqueness. This is done analogously to Remark 3.4. □

6. Conclusions and Future Work

We proved that all finitary endofunctors H generate a free iterative monad $\mathbb{R}$. All we needed in our proof was the assumptions that the base category is locally finitely presentable. This is the "real McCoy" that we tried to achieve in two preceeding papers [AMV$_1$] and [AMV$_2$]: there we obtained the same result only in the base category Set, and the proof was much more complicated. The reason was that when writing those papers we did not follow the footsteps of Evelyn Nelson and Jerzy Tiuryn who realized already more than twenty years ago that iterative algebras are more basic than Elgot's iterative theories.

The results of the present paper are analogous to results on completely iterative algebras and completely iterative theories. The latter were introduced in [EBT] in analogy to iterative theories by dropping the finiteness restriction on the objects of variables: one studies equation morphisms with arbitrary objects X of variables, and requests unique solutions of these more general equations. Stefan Milius [Mi] defines completely iterative algebras for a functor H on a category $\mathcal{A}$ with binary coproducts and he relates them to completely iterative monads: H has free completely iterative algebras TX if and only if H generates a free completely iterative monad T if and only if H has "enough final coalgebras", i. e., every functor $H(-) + X$ has a final coalgebra TX.

A natural question to ask, then, is whether there is a monad in between the free iterative monad R and the free completely iterative one T: how about considering, for an accessible functor and some uncountable cardinal λ, all equation morphisms with a λ-presentable object X of variables. We showed that the answer is negative: one gets the same monad, namely T, see Remark 5.15.

The main technical result of our paper is a description of an initial iterative algebra as a colimit of all H-coalgebras carried by finitely presentable objects. From this result we derived that the algebraic theory formed by all free iterative H-algebras is iterative in the sense of Calvin Elgot. In fact, that theory can be characterized as a free iterative theory on H. The freeness of the rational monad can be used to formulate clearly the "second-order substitution" described for rational Σ-trees by Bruno Courcelle [C], see Remark 5.14.

Our result can be applied to arbitrary base categories which are locally finitely presentable. For example, to the category of all finitary endofunctors of Set. In the future we intend to use this in an attempt to describe the monad of algebraic trees, see Courcelle [C], categorically.

References

[AAMV] P. Aczel, J. Adámek, S. Milius and J. Velebil, Infinite Trees and Completely Iterative Theories: A Coalgebraic View, *Theoret. Comput. Sci.* 300 (2003), 1–45.

[A$_1$] J. Adámek, Free Algebras and Automata Realization in the Language of Categories, *Comment. Math. Univ. Carolinae* 15 (1974), 589–602.

[A_2] J. Adámek, On a description of terminal coalgebras and iterative theories, *Electron. Notes Theor. Comput. Sci.* 82.1 (2003).

[AMV_1] J. Adámek, S. Milius and J. Velebil, Free Iterative Theories — A Coalgebraic View, *Math. Structures Comput. Sci.* 13 (2003), 259–320.

[AMV_2] J. Adámek, S. Milius and J. Velebil, On Rational Monads and Free Iterative Theories, *Electron. Notes Theor. Comput. Sci.* 69 (2003).

[AMV_3] J. Adámek, S. Milius and J. Velebil, Elgot Algebras: A Base for Denotational Semantics, submitted.

[AP] J. Adámek and H.-E. Porst, On Tree Coalgebras and Coalgebra Presentations, *Theoret. Comput. Sci.* 311 (2004), 257–283.

[AR] J. Adámek and J. Rosický, *Locally presentable and accessible categories*, Cambridge University Press, 1994.

[BM] J. Barwise and L. Moss, *Vicious Circles*, CSLI Public. Notes, vol. 60, Stanford, 1996.

[BE] S. Bloom and C. C. Elgot, The Existence and Construction of Free Iterative Theories, *J. Comput. System Sci.* 12 (1974), 305–318.

[C] B. Courcelle, Fundamental Properties of Infinite Trees, *Theoret. Comput. Sci.* 25 (1983), 95–169.

[E] C. C. Elgot, Monadic Computation and Iterative Algebraic Theories, in: *Logic Colloquium '73* (eds: H. E. Rose and J. C. Shepherdson), North-Holland Publishers, Amsterdam, 1975.

[EBT] C. C. Elgot, S. L. Bloom and R. Tindell, On the Algebraic Structure of Rooted Trees, *J. Comput. System Sci.* 16 (1978), 361–399.

[GU] P. Gabriel and F. Ulmer, *Lokal präsentierbare Kategorien*, Lecture N. Math 221, Springer-Verlag, Berlin 1971.

[GLM] N. Ghani, C. Lüth and F. De Marchi, *Electron. Notes Theor. Comput. Sci* 65(1), 2002.

[G] S. Ginali, Regular trees and the free iterative theory, *J. Comput. System Sci.* 18 (1979), 228–242.

[L] J. Lambek, A fixpoint theorem for complete categories, *Math. Z.* 103 (1968), 151–161.

[M] S. Mac Lane, *Categories for the Working Mathematician*, 2nd edition, Springer-Verlag, 1998.

[Mi] S. Milius, Completely Iterative Algebras and Completely Iterative Monads, accepted for publication in *Inform. and Comput.*

[N] E. Nelson, Iterative Algebras, *Theoret. Comput. Science* 25 (1983), 67–94.

[O] G. Osius, Categorical Set Theory: A characterization of the category of sets, *J. Pure Appl. Algebra* 4 (1974), 79–119.

[T] J. Tiuryn, Unique Fixed Points vs. Least Fixed Points, *Theoret. Comput. Sci.* 12 (1980), 229–254.

[TPT] Possible Origins of the Phrase *Real McCoy*, The Phrase Thesaurus, `http://phrases.shu.ac.uk`.

Institute of Theoretical Computer Science, Technical University, Braunschweig, Germany
E-mail address: {adamek,milius}@iti.cs.tu-bs.de

Faculty of Electrical Engineering, Technical University, Prague, Czech Republic
E-mail address: velebil@math.feld.cvut.cz

Elgot Algebras

Jiří Adámek[1,⋆], Stefan Milius[1], and Jiří Velebil[2,⋆]

[1] Institute of Theoretical Computer Science, Technical University, Braunschweig, Germany
{adamek,milius}@iti.cs.tu-bs.de
[2] Faculty of Electrical Engineering, Technical University, Prague, Czech Republic
velebil@math.feld.cvut.cz

If you are not part of the solution,
you are part of the problem.
Eldridge Cleaver, *speech in San Francisco*, 1968

Abstract. Denotational semantics can be based on algebras with additional structure (order, metric, etc.) which makes it possible to interpret recursive specifications. It was the idea of Calvin Elgot not to use additional structure and to base denotational semantics on iterative theories, i. e., theories in which abstract recursive specifications are required to have unique solutions. Later Bloom and Ésik studied iteration theories and iteration algebras in which a specified solution has to obey certain axioms. In this paper we propose so-called Elgot algebras. An Elgot algebra is an algebra with a specified solution for every system of flat recursive equations. That specification satisfies two simple and well motivated axioms: functoriality (stating that solutions are stable under renaming of recursion variables) and compositionality (stating how to perform simultaneous recursion). These two axioms stem canonically from Elgot's iterative theories: We prove that the category of Elgot algebras is the Eilenberg–Moore category of the monad given by a free iterative theory.

1 Introduction

This paper whose extended abstract was presented at the conference MFPS XXI, see [AMV$_3$], studies Elgot algebras, a new notion of algebra useful for application in the semantics of recursive computations. In programming, functions are often specified by a *recursive applicative program scheme* such as

$$\begin{aligned} \varphi(x) &\approx F(x, \varphi(Gx)) \\ \psi(x) &\approx F(\varphi(Gx), GGx) \end{aligned} \tag{1.1}$$

where F and G are given functions and φ and ψ are recursively defined in terms of the given ones by (1.1). We are interested in the semantics of such schemes. Actually, one has to distinguish between *uninterpreted* and *interpreted* semantics. In the uninterpreted semantics the givens are not functions but merely function symbols from a signature Σ. In the present paper we prepare a basis for the interpreted semantics in which a program scheme comes together with a suitable Σ-algebra A, which gives an interpretation to all the given function symbols. The actual application of Elgot algebras to semantics will be dealt with in [MM]. By "suitable algebra" we mean, of course, one in which recursive program schemes can be given a semantics. For example, for the recursive program scheme (1.1) we are only interested in those Σ-algebras A, where $\Sigma = \{ F, G \}$, in which the program scheme (1.1) has a *solution*, i. e., we can canonically obtain new operations φ^A and ψ^A on A so that the formal equations (1.1) become valid identities. The question we address is:

$$\text{What } \Sigma\text{-algebras are suitable for semantics?} \tag{1.2}$$

Several answers have been proposed in the literature. One well-known approach is to work with complete posets (CPO) in lieu of sets, see e.g. [GTWW]. Here algebras have an additional CPO

⋆ The first and the third author acknowledge the support of the Grant Agency of the Czech Republic under the Grant No. 201/02/0148.

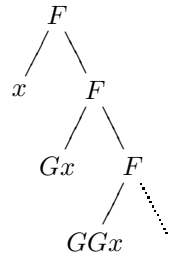
structure making all operations continuous. Another approach works with complete metric spaces, see e.g. [ARu]. Here we have an additional complete metric making all operations contracting. In both of these approaches one imposes extra structure on the algebra in a way that makes it possible to obtain the semantics of a recursive computation as a join (or limit, respectively) of finite approximations.

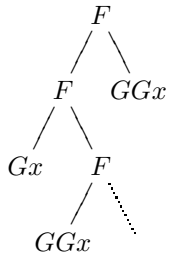
It was the idea of Calvin Elgot to try and work in a purely algebraic setting avoiding extra structure like order or metric. In [El] he introduced iterative theories which are algebraic theories in which certain systems of recursive equations have *unique* solutions. Later Evelyn Nelson [N] and Jerzy Tiuryn [T] studied iterative algebras, which are algebras for a signature Σ with unique solutions of recursive equations. While avoiding extra structure, these are still not the unifying concept one would hope for, since they do not subsume continuous algebras—least fixed points are typically not unique.

However, analyzing all the above types of algebras we find an interesting common feature which make continuous, metrizable and iterative algebras fit for use in semantics of recursive program schemes: these algebras allow for an interpretation of infinite Σ-trees. Let us make this more precise. For a given signature Σ consider the algebra

$$T_\Sigma X$$

of all (finite and infinite) Σ-trees over X, i. e., rooted ordered trees where inner nodes with n children are labelled by n-ary operation symbols from Σ, and leaves are labelled by constants or elements from X. The algebra $T_\Sigma X$ is the free continuous Σ-algebra on X and also the free metrizable Σ-algebra on X. Consequently, for any continuous or metrizable algebra A we obtain a canonical map $T_\Sigma A \longrightarrow A$ which provides for any Σ-tree over A its result of computation in A. It is then easy to give semantics to recursive program schemes in A. For example, for (1.1) one can simply take the tree unfolding which yields the infinite trees

$\varphi^\dagger(x) =$ 

$\psi^\dagger(x) =$ 

and then for any argument $x \in A$ compute these infinite trees in A.

Actually, we do not need to be able to compute all infinite trees: all recursive program schemes unfold to *algebraic trees*, see [C] (we mention these in the Summary shortly). Another important subclass are *rational trees*, which are obtained as all solutions of guarded finitary recursive equations. They were characterized in [G] as those Σ-trees having up to isomorphism finitely many subtrees only. We denote by

$$R_\Sigma X$$

the subalgebra of all rational trees in $T_\Sigma X$. With this in mind, we can restate problem (1.2) more formally:

$$\begin{array}{c}\text{What } \Sigma\text{-algebras have a suitable computation of all trees?}\\ \text{Or all rational trees?}\end{array} \tag{1.3}$$

This means, one further step more formally: what is the largest category of Σ-algebras in which $T_\Sigma X$, or $R_\Sigma X$, respectively, act as free algebras on X? The answer in case of $T_\Sigma X$ is: complete Elgot algebras. These are Σ-algebras A with an additional operation "dagger" assigning to every system e of recursive equations in A a solution $e^\dagger$. Two (surprisingly simple) axioms are put on $(-)^\dagger$ which stem from the internal structure of $T_\Sigma X$: the functor T_Σ given by $X \longmapsto T_\Sigma X$ is part of a monad in Set, and this is the free completely iterative theory on Σ, as proved in [EBT].

We will prove that the monadic algebras of this monad (i. e., the Eilenberg–Moore category of T_Σ) is precisely the category of complete Elgot algebras. Basic examples: continuous algebras or metrizable algebras are Elgot algebras. Analogously, the largest category of Σ-algebras in which each $R_\Sigma X$ acts as a free algebra are Elgot algebras. They are defined precisely as the complete Elgot algebras, except that the systems e of recursive equations considered there are required to be finite. For example, every iterative algebra is an Elgot algebra.

Related Work: Solutions of recursive equations are a fundamental part of a number of models of computation, e. g., iterative theories of C. Elgot [El], iteration theories of S. Bloom and Z. Ésik [BÉ], traced monoidal categories of A. Joyal, R. Street and D. Verity [JSV], fixed-point theories for domains, see S. Eilenberg [Ei] or G. Plotkin [P], etc. In some of these models the assignment of a solution $e^\dagger$ to a given type of recursive equation e is unique (e. g., in iterative theories every ideal system has a unique solution, or in domains given by a complete metric space there are unique solutions of fixed-point equations, see [ARu]). The operation $e \longmapsto e^\dagger$ then satisfies a number of equational properties. In other models, e. g., in iteration theories or in the traced cartesian categories, see [Ha], a specific choice of a solution $e^\dagger$ is assumed, and certain properties (inspired by the models with unique solutions) are formulated as axioms.

The approach of the present paper is more elementary in asking for solutions $e \longmapsto e^\dagger$ in a concrete algebra A. Here we work with flat equations e in A, i. e., morphisms of the form $e : X \longrightarrow HX + A$, but flatness is just a technical restriction: in future research we will prove that more general non-flat equations obtain solutions "automatically". The fact that we work with a fixed algebra A (and let only X and e vary) is partly responsible for the simplicity of our axioms in comparison to the work on theories (where A varies as well), see e. g. [BÉ] or [SP_1]. Iterative algebras of Evelyn Nelson [N] and Jerzy Tiuryn [T], where solutions $e^\dagger$ are required to be unique, are a similar approach. And iteration algebras of Zoltan Ésik [É] are another one. Unfortunately, the number of axioms (seven) and their complexity make the question of the relationship of that notion to Elgot algebras a nontrivial one. We intend to study this question in the future.

We work with two variations: Elgot algebras, related to $R_\Sigma X$, where the function $(-)^\dagger$ assigns a solution only to finitary flat recursive systems, and complete Elgot algebras, related to $T_\Sigma X$, where the function $(-)^\dagger$ assigns solutions to all flat recursive systems. This is related to our previous research [AAMV,M,AMV_1,AMV_2] in which we proved that every finitary endofunctor H generates a free iterative monad R, and a free completely iterative monad T. In the present paper we then study the Eilenberg–Moore categories of the monads R and T. Here H is an endofunctor of a category satisfying some rather mild conditions (not only Set): this generality does not make the proofs any more complex, and later we use other categories than Set (see Summary).

2 Iterative Algebras and CIAs

Assumption 2.1. Throughout the paper H denotes an endofunctor of a category $\mathcal{A}$ having binary coproducts. We denote by $\mathsf{inl} : A \longrightarrow A+B$ and $\mathsf{inr} : B \longrightarrow A+B$ the corresponding injections. At some stage we assume that $\mathcal{A}$ is locally finitely presentable and that H is finitary, i. e., preserves filtered colimits, but we then make these assumptions explicitly.

Recall that an object X is called *finitely presentable* iff the hom-functor $\mathcal{A}(X, -) : \mathcal{A} \longrightarrow \mathsf{Set}$ is finitary. (In Set, these are precisely the finite sets. In equational classes of algebras these are precisely the finitely presentable algebras in the usual sense.) Recall further that a category $\mathcal{A}$ is called *locally finitely presentable* if it has colimits and a small collection of finitely presentable objects whose closure under filtered colimits is all of $\mathcal{A}$, see [AR].

Definition 2.2. *Let $\alpha : HA \longrightarrow A$ be an H-algebra. By a* flat equation morphism *in A we understand a morphism $e : X \longrightarrow HX + A$ in $\mathcal{A}$. We call e* finitary *provided that X is finitely*

presentable. A solution *of e is a morphism $e^\dagger : X \longrightarrow A$ such that the square*

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha, A]} \\ HX + A & \xrightarrow[He^\dagger + A]{} & HA + A \end{array} \tag{2.1}$$

commutes.

If every finitary flat equation morphism has a unique solution, then A is said to be an iterative algebra*. And A is called a* completely iterative algebra *(CIA) if every flat equation morphism has a unique solution.*

Remark 2.3. Iterative algebras of polynomial endofunctors of Set were introduced and studied by Evelyn Nelson [N]. She proved that the algebras $R_\Sigma X$ of rational Σ-trees on X form free iterative algebras, and that the theory obtained from them is a free iterative theory of Calvin Elgot [El]. We have recently studied iterative algebras in a much more general setting; working with a finitary endofunctor of a locally finitely presentable category. Completely iterative algebras were studied by Stefan Milius in [M].

Example 2.4. Consider algebras in Set with one binary operation $*$, i. e., the functor is $HX = X \times X$. A flat equation morphism e in an algebra A assigns to every variable x either a flat term $y * z$ (y and z are variables) or an element of A. A solution $e^\dagger : X \longrightarrow A$ assigns to $x \in X$ either the same element as e, in case $e(x) \in A$, or the result of $e^\dagger(y) * e^\dagger(z)$, in case $e(x) = y * z$. For example, the following recursive equation

$$x \approx x * x\,,$$

represented by the obvious morphism $e : \{ x \} \longrightarrow \{ x \} \times \{ x \} + A$, has as solution $e^\dagger$ an element $a = e^\dagger(x)$ which is idempotent. Consequently, every iterative algebra has a unique idempotent. If A is even completely iterative, then it has, for each sequence $a_0, a_1, a_2, \ldots$ of elements, a unique interpretation of $a_0 * (a_1 * (a_2 \cdots)))$, i. e., a unique sequence $b_0, b_1, b_2, \ldots$ with $b_0 = a_0 * b_1$, $b_1 = a_1 * b_2$, etc. In fact, we consider here the equations

$$x_n \approx a_n * x_{n+1} \qquad (n \in \mathbb{N})\,.$$

Iterative algebras have unique solutions of many non-flat equations because we can flatten them. For example the following recursive equations

$$x_1 \approx (x_2 * a) * b \qquad x_2 \approx x_1 * b$$

are not flat. But they can be easily flattened to obtain a system

$$\begin{array}{ll} x_1 \approx z_1 * z_2 & x_2 \approx x_1 * z_2 \\ z_1 \approx x_2 * z_3 & z_2 \approx b \\ z_3 \approx a & \end{array}$$

represented by a morphism $e : X \longrightarrow X \times X + A$, where $X = \{ x_1, x_2, z_1, z_2, z_3 \}$. Its solution is a map $e^\dagger : X \longrightarrow A$ yielding a pair of elements $s = e^\dagger(x_1)$ and $t = e^\dagger(x_2)$ satisfying $s = (t * a) * b$ and $t = s * a$.

Example 2.5. Iterative Σ-algebras. For every finitary signature $\Sigma = (\Sigma_n)_{n \in \mathbb{N}}$ we can identify Σ-algebras with algebras of the *polynomial endofunctor* H_Σ of Set defined on objects X by

$$H_\Sigma X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X \times X + \ldots$$

A Σ-term which has the form $\sigma(x_1, \ldots, x_k)$ for some $\sigma \in \Sigma_k$ and for variables $x_1, \ldots, x_k$ from X is called *flat*. Then a flat equation morphism $e : X \longrightarrow H_\Sigma X + A$ in an algebra A represents a system

$$x \approx t_x$$

of recursive equations, one for every variable $x \in X$, where each t_x is either a flat term in X, or an element of A. A solution $e^\dagger$ assigns to every variable x with $t_x = a$, $a \in A$, the element a, and if $t_x = \sigma(x_1, \ldots, x_k)$ then $e^\dagger(x) = \sigma_A(e^\dagger(x_1), \ldots, e^\dagger(x_k))$.

Observe that every iterative Σ-algebra A has, for every $\sigma \in \Sigma_k$, a unique idempotent (i.e., a unique element $a \in A$ with $\sigma(a, \ldots, a) = a$). In fact, consider the flat equation $x \approx \sigma(x, \ldots, x)$. More generally, every Σ-polynomial has a unique idempotent in A. For example, for a polynomial of depth 2, $\sigma(\tau_1, \ldots, \tau_k)$, where $\sigma \in \Sigma_k$ and $\tau_1, \ldots, \tau_k \in \Sigma_n$ consider the recursive equations

$$\begin{aligned} x_0 &\approx \sigma(x_1, x_2, \ldots, x_k) \\ x_i &\approx \tau_i(x_0, x_0, \ldots, x_0) \qquad (i = 1, \ldots, k)\,. \end{aligned}$$

An example of an iterative Σ-algebra is the algebra T_Σ of all (finite and infinite) Σ-trees. Also the subalgebra R_Σ of T_Σ of all rational Σ-trees is iterative, see [N].

Example 2.6. In particular, for unary algebras ($H = Id$), an algebra $\alpha : A \longrightarrow A$ is iterative iff α^k has a unique fixed point ($k \geq 1$), see [AMV$_2$]. And A is a CIA iff, moreover, there exists no infinite sequence $(a_n)_{n\in\mathbb{N}}$ in $\mathcal{A}$ with $\alpha a_{n+1} = a_n$, see [M].

Remark 2.7. In [AMV$_2$] we have proved that for every finitary functor H of a locally finitely presentable category $\mathcal{A}$ a free iterative algebra RY exists on every object Y. Furthermore, we have given a canonical construction of RY as a colimit of all coalgebras $X \longrightarrow HX + Y$ carried by finitely presentable objects, in other words, for every object Y of $\mathcal{A}$, RY is a colimit of all finitary flat equations in Y. For example, for a polynomial functor H_Σ of Set the free iterative algebra on a set Y is the algebra $R_\Sigma Y$ of all rational Σ-trees over Y. In general, we call the monad $\mathbb{R}$ of free iterative algebras the *rational monad* generated by H. We have proved in [AMV$_2$] that the rational monad $\mathbb{R}$ is a free iterative monad on H.

Example 2.8. Completely metrizable algebras. Complete metric spaces are well-known to be a suitable basis for semantics. The first categorical treatment of complete metric spaces for semantics is due to P. America and J. Rutten [ARu]. Let

$$\mathsf{CMS}$$

denote the category of all complete metric spaces (i.e., such that every Cauchy sequence has a limit) with metrics in the interval $[0, 1]$. The morphisms are nonexpanding maps $f : (X, d_X) \longrightarrow (Y, d_Y)$, i. e., the inequality $d_Y(f(x), f(x')) \leq d_X(x, x')$ holds for all x, x' in X.

Given complete metric spaces X and Y, the hom-set $\mathsf{CMS}(X, Y)$ carries the pointwise metric $d_{X,Y}$ defined as follows:

$$d_{X,Y}(f, g) = \sup_{x\in X} d_Y(f(x), g(x))$$

America and Rutten call a functor $H : \mathsf{CMS} \longrightarrow \mathsf{CMS}$ *contracting* if there exists a constant $\varepsilon < 1$ such that for arbitrary morphisms $f, g : X \longrightarrow Y$ we have

$$d_{HX,HY}(Hf, Hg) \leq \varepsilon \cdot d_{X,Y}(f, g).$$

Lemma 2.9. *If $H : \mathsf{CMS} \longrightarrow \mathsf{CMS}$ is a contracting functor, then every nonempty H-algebra is a CIA.*

Proof. Let $\alpha : HA \longrightarrow A$ be a nonempty H-algebra. Choose an element a of A. For every equation morphism $e : X \longrightarrow HX + A$ define a sequence $e^\dagger_n$ in $\mathsf{CMS}(X, A)$ as follows:

1. $e_0^\dagger = \mathsf{const}_a$, the constant function of value a.
2. Given $e_n^\dagger$ then $e_{n+1}^\dagger$ is defined as follows (compare (2.1)):

$$\begin{array}{ccc} X & \xrightarrow{e_{n+1}^\dagger} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha, A]} \\ HX + A & \xrightarrow[He_n^\dagger + A]{} & HA + A \end{array} \tag{2.2}$$

We prove that $(e_n^\dagger)$ is a Cauchy sequence in $\mathsf{CMS}(X, A)$. In fact, put

$$z = d({e_0}^\dagger, {e_1}^\dagger),$$

then we prove by induction that

$$d(e_n^\dagger, e_{n+1}^\dagger) \leq z \cdot \varepsilon^n.$$

For the induction step from the above inequality derive $d(He_{n+1}^\dagger, He_n^\dagger) \leq z \cdot \varepsilon^{n+1}$ and then use the definition of $e_n^\dagger$ and $e_{n+1}^\dagger$, see (2.2). Consequently, the sequence $(e_n^\dagger)$ is Cauchy: for every number $\delta > 0$ choose k with $z \cdot \varepsilon^k < \delta \cdot (1 - \varepsilon)$. Then for all n the inequalities

$$d({e_k}^\dagger, {e_{k+n}}^\dagger) \leq \sum_{i=0}^{n-1} d(e_{k+i}^\dagger, e_{k+i+1}^\dagger) \leq z \cdot \sum_{i=0}^{n-1} \varepsilon^{k+i} < z \cdot \varepsilon^k \cdot \sum_{i=0}^{\infty} \varepsilon^i = \frac{z \cdot \varepsilon^k}{1 - \varepsilon} < \delta$$

take place. Consequently, a limit

$$e^\dagger = \lim_{n \to \infty} e_n^\dagger$$

exists in $\mathsf{CMS}(X, A)$. Due to the contractivity of H, it follows that $He^\dagger = \lim_{n \to \infty} He_n^\dagger$ in $\mathsf{CMS}(HX, HA)$ and thus the equality $He^\dagger + id_A = \lim_{n \to \infty} (He_n^\dagger + id_A)$ holds in $\mathsf{CMS}(HX + A, HA + A)$. Thus, $e^\dagger$ is a solution of e:

$$\begin{aligned} [\alpha, A] \cdot (He^\dagger + A) \cdot e &= \lim_{n \to \infty} [\alpha, A] \cdot (He_n^\dagger + A) \cdot e \\ &= \lim_{n \to \infty} e_{n+1}^\dagger \\ &= e^\dagger. \end{aligned}$$

Let $e^* : X \longrightarrow A$ be another solution of e. Put $b = d(e^\dagger, e^*)$. Then $d(He^\dagger, He^*) \leq \varepsilon \cdot b$ which implies $d(He^\dagger + id_A, He^* + id_A) \leq \varepsilon \cdot b$, consequently,

$$b = d(e^\dagger, e^*) = d([\alpha, A] \cdot (He^\dagger + A) \cdot e, [\alpha, A] \cdot (He^* + A) \cdot e) \leq \varepsilon \cdot b.$$

Since $\varepsilon < 1$, this implies $b = 0$. Thus, $e^\dagger$ is the unique solution.

Remark 2.10. Many set functors H have a lifting to contracting endofunctors H' of CMS. That is, for the forgetful functor $U : \mathsf{CMS} \longrightarrow \mathsf{Set}$ the following square

$$\begin{array}{ccc} \mathsf{CMS} & \xrightarrow{H'} & \mathsf{CMS} \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathsf{Set} & \xrightarrow[H]{} & \mathsf{Set} \end{array}$$

commutes. For example, if $HX = X^n$, define

$$H'(X, d) = (X^n, \frac{1}{2} \cdot d')$$

(where d' is the maximum metric) which is a contracting functor with $\varepsilon = \frac{1}{2}$. Since coproducts of $\frac{1}{2}$-contracting liftings are $\frac{1}{2}$-contracting liftings of coproducts, we conclude that every polynomial endofunctor has a contracting lifting to CMS.

Let us call an H-algebra $\alpha : HA \longrightarrow A$ *completely metrizable* if there exists a complete metric, d, on A such that α is a nonexpanding map from $H'(A,d)$ to (A,d).

Corollary 2.11. *Every completely metrizable algebra A is a CIA.*

In fact, to every equation morphism $e : X \longrightarrow HX + A$ assign the unique solution of $e : (X, d_0) \longrightarrow H'(X, d_0) + (A, d)$, where d_0 is the discrete metric ($d_0(x,x') = 1$ iff $x \neq x'$).

Remark 2.12. Stefan Milius [M] proved that for any endofunctor H of $\mathcal{A}$ a final coalgebra TY of $H(-) + Y$ is a free CIA on Y, and conversely. Furthermore, assuming that the free CIAs exist, it follows that the monad $\mathbb{T}$ of free CIAs is a free completely iterative monad on H. This generalizes and extends the classical result of [EBT] since for a polynomial functor H_Σ of Set the free completely iterative algebra on a set Y is the algebra $T_\Sigma Y$ of all Σ-trees over Y.

Remark 2.13. We are going to prove two properties of iterative algebras and CIA's: the functoriality and compositionality for solutions. We will use two "operations" on equation morphisms. One, $\bullet$, is just change of parameter names: given a flat equation morphism $e : X \longrightarrow HX + Y$ and a morphism $h : Y \longrightarrow Z$ we obtain the following equation morphism

$$h \bullet e \equiv X \xrightarrow{\ e\ } HX + Y \xrightarrow{\ HX+h\ } HX + Z\,.$$

The other operation $\blacksquare$ combines two flat equation morphisms

$$e : X \longrightarrow HX + Y \qquad \text{and} \qquad f : Y \longrightarrow HY + A$$

into the single flat equation morphism $f \blacksquare e : X + Y \longrightarrow H(X + Y) + A$ in a canonical way: put $\mathsf{can} = [H\mathsf{inl}, H\mathsf{inr}] : HX + HY \longrightarrow H(X + Y)$ and define

$$f \blacksquare e \equiv X+Y \xrightarrow{[e,\mathsf{inr}]} HX+Y \xrightarrow{HX+f} HX+HY+A \xrightarrow{\mathsf{can}+A} H(X+Y)+A, \tag{2.3}$$

2.14. Functoriality. This states that solutions are invariant under renaming of variables, provided, of course, that the right-hand sides of equations are renamed accordingly. Formally, observe that every flat equation morphism is a coalgebra of the endofunctor $H(-)+A$. Given two such coalgebras e and f, a renaming of the variables (or *morphism of equations*) is a morphism $h : X \longrightarrow Y$ which forms a coalgebra homomorphism:

$$\begin{array}{ccc} X & \xrightarrow{\ e\ } & HX + A \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh+A} \\ Y & \xrightarrow[\ f\ ]{} & HY + A \end{array} \tag{2.4}$$

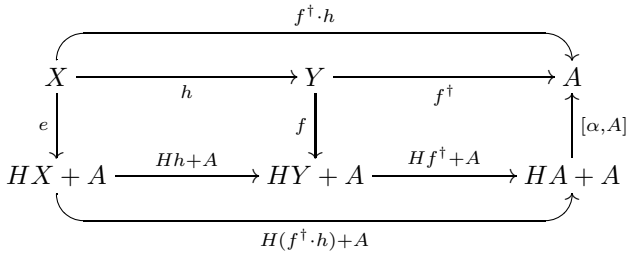
Definition 2.15. *Let A be an algebra with a choice $e \longmapsto e^\dagger$ of solutions, for all flat equation morphisms e in A. We say that the choice is* functorial *provided that*

$$e^\dagger = f^\dagger \cdot h \tag{2.5}$$

holds for all equation morphisms $h : e \longrightarrow f$. In other words: $(-)^\dagger$ is a functor from the category of all flat equation morphisms in the algebra A into the comma-category of the object A.

Lemma 2.16. *In every CIA the assignment $(-)^\dagger$ is functorial.*

Proof. For each morphism h of equations the diagram



commutes. Thus, $f^\dagger \cdot h$ is a solution of e. Uniqueness of solutions now implies the desired result.

Remark. The same holds for every iterative algebra, except that there we restrict X and Y in Definition 2.15 to finitely presentable objects.

2.17. Compositionality. This tells us how to perform simultaneous recursion: given an equation morphism f in A with a variable object Y, we can combine it with any equation morphism e in Y with a variable object X to obtain the equation morphism $f \blacksquare e$ in A of Remark 2.13. The compositionality decrees that the left-hand component of $(f \blacksquare e)^\dagger$ is just the solution of $f^\dagger \bullet e$, i. e., in lieu of solving f and e simultaneously we first solve f, plug in the solution in e and solve the resulting equation morphism.

Definition 2.18. *Let A be an algebra with a choice $e \longmapsto e^\dagger$ of solutions, for all flat equation morphisms e in A. We say that the choice is* compositional *if for each pair $e : X \longrightarrow HX + Y$ and $f : Y \longrightarrow HY + A$ of flat equation morphisms the equation below holds.*

$$(f^\dagger \bullet e)^\dagger = (f \blacksquare e)^\dagger \cdot \mathsf{inl} \tag{2.6}$$

Remark 2.19. Notice that the coproduct injection $\mathsf{inr} : Y \longrightarrow X + Y$ is a morphism of equations from f to $f \blacksquare e$. Functoriality then implies that $f^\dagger = (f \blacksquare e)^\dagger \cdot \mathsf{inr}$. Thus, in the presence of functoriality, the compositionality is equivalent to

$$(f \blacksquare e)^\dagger = [(f^\dagger \bullet e)^\dagger, f^\dagger]\,. \tag{2.7}$$

Lemma 2.20. *In every CIA the assignment $(-)^\dagger$ is compositional.*

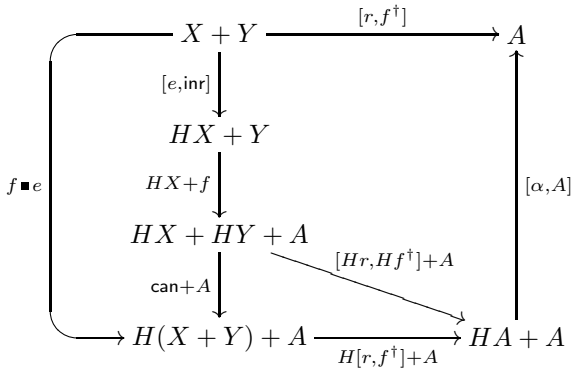
Proof. Denote by

$$r = (f^\dagger \bullet e)^\dagger : X \longrightarrow A$$

the solution of $f^\dagger \bullet e$. It is sufficient to prove that

$$(f \blacksquare e)^\dagger = [r, f^\dagger] : X + Y \longrightarrow A\,.$$

That is, by the uniqueness of solutions, that the following square



(2.8)

commutes. This is clear for the right-hand components (with domain Y):

$$[\alpha, A] \cdot ([Hr, Hf^\dagger] + A) \cdot \mathsf{inr} \cdot f = [\alpha, A] \cdot (Hf^\dagger + A) \cdot f = f^\dagger$$

because $f^\dagger$ solves f. For the left-hand components (with domain X) use the commutativity of the square defining $r = (f^\dagger \bullet e)^\dagger$:

$$\begin{array}{ccc} X & \xrightarrow{\quad r \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha,A]} \\ HX + Y & & \\ {\scriptstyle HX+f^\dagger}\downarrow & & \\ HX + A & \xrightarrow[Hr+A]{} & HA + A \end{array} \tag{2.9}$$

We now only need to show that the passages from $HX+Y$ to A in the above squares (2.8) and (2.9) are equal. The left-hand components are, in both cases, $\alpha \cdot Hr : HX \longrightarrow A$. For the right-hand components use $f^\dagger = [\alpha, A] \cdot (Hf^\dagger + A) \cdot f$.

Remark 2.21. The same holds for every iterative algebra, except that here we restrict X and Y in Definition 2.18 to finitely presentable objects.

Remark 2.22. As mentioned in the Introduction, our two axioms, functoriality and compositionality, are not new as ideas of axiomatizing recursion—we believe however, that their concrete form is new, and their motivation strengthened by the results below.

Functoriality corresponds precisely to the "functorial dagger implication" of S. Bloom and Z. Ésik [BÉ], 5.3.3, which states that for every object p of an iterative theory the formation $f \longmapsto f^\dagger$ of solutions for ideal morphisms $f : m \longrightarrow m + p$ is a functor. And the compositionality resembles the "left pairing identity" of [BÉ], 5.3.1, which for $f : n \longrightarrow n + m + p$ and $g : m \longrightarrow n + m + p$ states that

$$[f, g]^\dagger = [f^\dagger \cdot [h^\dagger, id_p], h^\dagger]\,,$$

where

$$h \equiv\ m \xrightarrow{\ g\ } n + m + p \xrightarrow{\ [f^\dagger, id_{m+p}]\ } m + p\,.$$

This identity corresponds also to the Bekić-Scott identity, see e. g. [Mo], 2.1.

3 Elgot Algebras

Definition 3.1. *Let H be an endofunctor of a category with finite coproducts. An* Elgot algebra *is an H-algebra $\alpha : HA \longrightarrow A$ together with a function $(-)^\dagger$ which to every finitary flat equation morphism*

$$e : X \longrightarrow HX + A \qquad (X \text{ finitely presentable})$$

assigns a solution $e^\dagger : X \longrightarrow A$ in such a way that the functoriality (2.5) and the compositionality (2.6) are satisfied.

By a complete Elgot algebra *we analogously understand an H-algebra together with a function $(-)^\dagger$ assigning to every flat equation e a solution $e^\dagger$ so that functoriality and compositionality are satisfied.*

Example 3.2. Every join semilattice A is an Elgot algebra. More precisely: consider the polynomial endofunctor $HX = X \times X$ of Set (expressing one binary operation). Then for every join semilattice A there is a "canonical" structure of an Elgot algebra on A obtained as follows: the algebra RA of all rational binary trees on A has an interpretation on A given by the function $\alpha : RA \longrightarrow A$ forming, for every rational binary tree t the join of all the (finitely many!) labels of leaves of t in A. Now given a finitary flat equation morphism $e : X \longrightarrow X \times X + A$, it has a unique solution $e^\dagger : X \longrightarrow RA$ in the free iterative algebra RA, and composed with α this yields a structure $e \longmapsto \alpha \cdot e^\dagger$ of an Elgot algebra on A. See Example 4.9 for a proof.

Remark 3.3. In contrast, no nontrivial join semilattice is iterative. In fact, in an iterative join semilattice there must be a unique solution of the formal equation $x \approx x \vee x$.

Example 3.4. Continuous algebras on cpos are complete Elgot algebras. Let us work here in the category

$$\mathsf{CPO}$$

of all *ω-complete posets*, i.e., posets having joins of increasing ω-chains; morphisms are the *continuous functions*, i.e., functions preserving joins of ω-chains. A functor $H : \mathsf{CPO} \longrightarrow \mathsf{CPO}$ is called *locally continuous* provided that for arbitrary CPOs, X and Y, the derived function from $\mathsf{CPO}(X,Y)$ to $\mathsf{CPO}(HX,HY)$ is continuous (i.e., $H(\bigsqcup f_n) = \bigsqcup Hf_n$ holds for all increasing ω-sequences $f_n : X \longrightarrow Y$). For example, every polynomial endofunctor $X \longmapsto \coprod_n \Sigma_n \times X^n$ of CPO (where Σ_n are cpos) is locally continuous.

Observe that the category CPO has coproducts: they are the disjoint unions with elements of different summands incompatible.

Proposition 3.5. *Let $H : \mathsf{CPO} \longrightarrow \mathsf{CPO}$ be a locally continuous functor and let $\alpha : HA \longrightarrow A$ be an H-algebra with a least element $\bot \in A$. Then $(A, \alpha, (-)^\dagger)$ is a complete Elgot algebra w.r.t. the assignment of the least solution $e^\dagger$ to every flat equation morphism e.*

Remark. Notice that the least solution of $e : X \longrightarrow HX + A$ refers to the elementwise order of the hom-set $\mathsf{CPO}(X,A)$. We can actually prove a concrete formula for $e^\dagger$ as a join of the ω-chain

$$e^\dagger = \bigsqcup_{n\in\omega} e^\dagger_n$$

of "approximations": $e^\dagger_0$ is the constant function to $\bot$, the least element of A, and given $e^\dagger_n$, then $e^\dagger_{n+1}$ is defined by the commutativity of (2.2).

Proof. (1) Since $e^\dagger_0$ is the least element of $\mathsf{CPO}(X,A)$, we have $e^\dagger_0 \sqsubseteq e^\dagger_1$. Since H, being locally continuous, is locally order-preserving, it follows by easy induction that the chain $(e^\dagger_n)$ is increasing in $\mathsf{CPO}(X,A)$. Consequently, the join $e^\dagger = \bigsqcup e^\dagger_n$ exists. The commutative diagrams (2.2) yield the diagram (2.1) showing that $e^\dagger$ is in fact a solution of e.

(2) $e^\dagger$ is the smallest solution. In fact, given a solution

$$\begin{array}{ccc} X & \xrightarrow{\quad e^*\quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha,A]} \\ HX + A & \xrightarrow[He^*+A]{} & HA + A \end{array}$$

we prove $e^\dagger_n \sqsubseteq e^*$ by induction on n, then $e^\dagger \sqsubseteq e^*$. The case $n = 0$ is clear. Given $e^\dagger_n \sqsubseteq e^*$, then $He^\dagger_n \sqsubseteq He^*$ (due to the local continuity of H) which implies

$$e^\dagger_{n+1} = [\alpha, A] \cdot (He^\dagger_n + A) \cdot e \sqsubseteq [\alpha, A] \cdot (He^* + A) \cdot e = e^* \,.$$

(3) The assignment $e \longmapsto e^\dagger$ is functorial. In fact, let

$$\begin{array}{ccc} X & \xrightarrow{\quad e\quad} & HX + A \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh+A} \\ Y & \xrightarrow[f]{} & HY + A \end{array}$$

be a coalgebra homomorphism. It is easy to see by induction that

$$e^\dagger_n = f^\dagger_n \cdot h \quad (\text{for all } n \geq 0)\,,$$

thus, $e^\dagger = f^\dagger \cdot h$.

(4) We prove the compositionality. Let

$$e : X \longrightarrow HX + Y \quad \text{and} \quad f : Y \longrightarrow HY + A$$

be given. We shall show that the equality

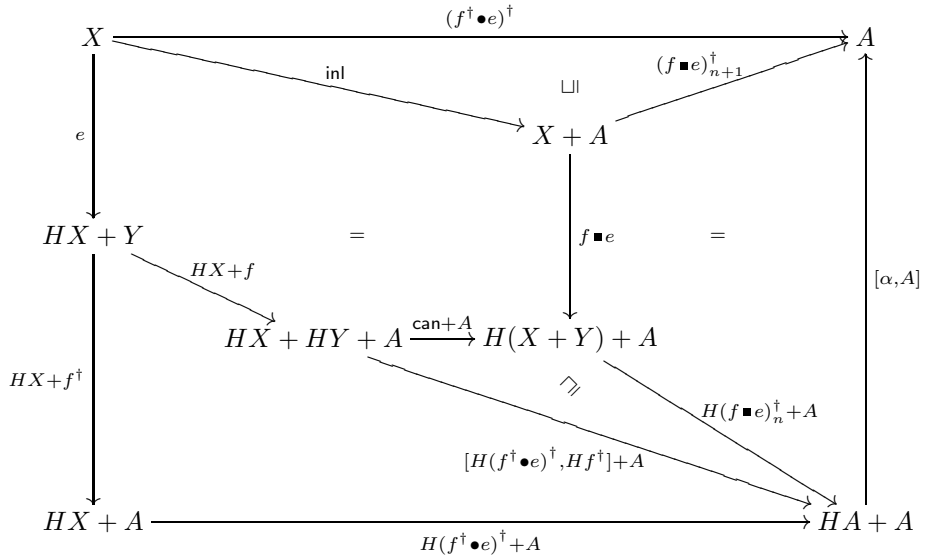
$$(f \blacksquare e)^\dagger \cdot \mathsf{inl} = (f^\dagger \bullet e)^\dagger$$

holds. It suffices to prove, by induction on n, that the following two inequalities

$$(f \blacksquare e)^\dagger_n \cdot \mathsf{inl} \sqsubseteq (f^\dagger \bullet e)^\dagger \tag{3.1}$$

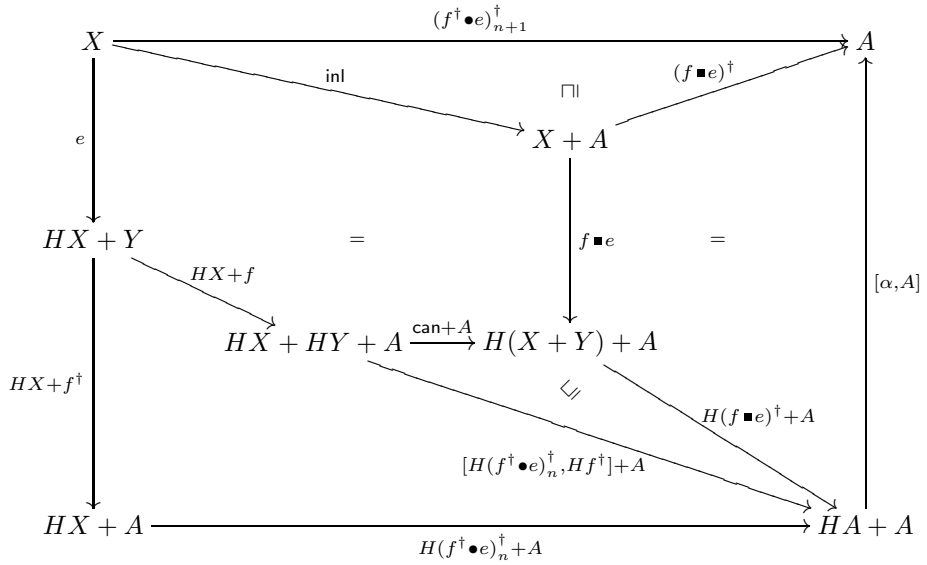
$$(f^\dagger \bullet e)^\dagger_n \sqsubseteq (f \blacksquare e)^\dagger \cdot \mathsf{inl} \tag{3.2}$$

hold. Observe first that $\mathsf{inr} : (Y, f) \longrightarrow (X + Y, f \blacksquare e)$ is a coalgebra homomorphism. Thus, the equation $(f \blacksquare e)^\dagger \cdot \mathsf{inr} = f^\dagger$ holds by functoriality. For the induction step for (3.1) consider the following diagram



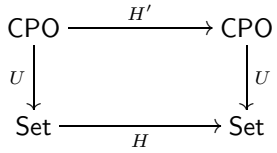
In order to prove the desired inequality in the upper triangle, we use the fact that the outward square commutes by definition of $(-)^\dagger$. The three middle parts clearly behave as indicated (for the triangle use the induction hypothesis (3.1) and $(f \blacksquare e)^\dagger_n \cdot \mathsf{inr} \sqsubseteq (f \blacksquare e)^\dagger \cdot \mathsf{inr} = f^\dagger$), And the lowest part commutes when extended by $[\alpha, A]$: In fact, for the left-hand component with domain HX this is trivial; for the right-hand component with domain Y use $f^\dagger = [\alpha, A] \cdot (Hf^\dagger + A) \cdot f$, see (2.1).

For the induction step for (3.2) consider the following diagram



The outer square commutes by definition of $(f^\dagger \bullet e)^\dagger_{n+1}$. The three middle parts behave as indicated (for the inequality use the induction hypothesis), and the lowest part commutes when extended by $[\alpha, A]$ as before. Thus, we obtain the desired inequality in the upper triangle.

Remark 3.6. Many set functors H have a lifting to locally continuous endofunctors H' of CPO. That is, for the forgetful functor $U : \mathsf{CPO} \longrightarrow \mathsf{Set}$ the following square



commutes. For example, every polynomial functor H_Σ has such a lifting. Let us call an H-algebra $\alpha : HA \longrightarrow A$, CPO-*enrichable* if there exists a CPO-ordering $\sqsubseteq$ with a least element on the set A such that α is a continuous function from $H'(A, \sqsubseteq)$ to $(A, \sqsubseteq)$.

Corollary 3.7. *Every* CPO*-enrichable* H*-algebra* A *in* Set *is a complete Elgot algebra.*

In fact, to every equation morphism $e : X \longrightarrow HX + A$ assign the least solution of $e : (X, \leq) \longrightarrow H'(X, \leq) + (A, \sqsubseteq)$ where $\leq$ is the discrete ordering of X ($x \leq y$ iff $x = y$).

Example 3.8. Unary algebras. Let $H = Id$ as an endofunctor of Set. Given an H-algebra $\alpha : A \longrightarrow A$, if α has no fixed point, then A carries no structure of an Elgot algebra: consider the equation $x \approx \alpha(x)$.

Conversely, every fixed point a_0 of α yields a flat cpo structure with a least element a_0 on A, i. e., $x \leq y$ iff $x = y$ or $x = a_0$. Thus, A is a complete Elgot algebra since it is CPO-enrichable. Notice that for every flat equation morphism $e : X \longrightarrow X + A$ the least solutions $e^\dagger$ operates as follows: for a variable x we have

$$e^\dagger(x) = \begin{cases} \alpha^k(a) & \text{if there is a sequence } x = x_0, x_1, \ldots x_k \text{ in } X \text{ that} \\ & \text{fulfils } e(x_0) = x_1, \ldots e(x_{k-1}) = x_k \text{ and } e(x_k) = a \\ a_0 & \text{else}. \end{cases}$$

Remark 3.9. For unary algebras, Example 3.8 describes *all* existing Elgot algebras. In fact, let $(A, \alpha, (-)^\dagger)$ be an Elgot algebra and let a_0 be the chosen solution of $x \approx \alpha(x)$ (i. e., of $\mathsf{inl} : 1 \longrightarrow 1 + A$). Then for every flat equation morphism $e : X \longrightarrow X + A$ the chosen solution sends a variable $x \in X$ to one of the above values $\alpha^k(a)$ or a_0. To prove this denote by $Y \subseteq X$ the set of all variables for which the "else" case holds above (i. e., no sequence $x = x_0, \dots x_k$ in X fulfils $e(x_i) = x_{i+1}$, for $i = 0, \dots, k-1$, and $e(x_k) \in A$). Apply functoriality to the morphism h from e to $1 + e : 1 + X \longrightarrow 1 + X + A$ defined by $h(y) \in 1$ for $y \in Y$ and $h(x) = x \in X$ else. In fact, the chosen solution of the unique element of 1 in $1 + X$ must be a_0 by functoriality (consider the left-hand coproduct injection from the flat equation morphism $\mathsf{inl} : 1 \longrightarrow 1 + A$ to $1 + e$).

Example 3.10. Every complete lattice A is a complete Elgot algebra of $HX = X \times X$. Analogously to Example 3.2 we have a function $\alpha : TA \longrightarrow A$ assigning to every binary tree t in TA the join of all labels of leaves of t in A. Now for every flat equation morphism e in A we have its unique solution $e^\dagger$ in TA and this yields a structure $e \longmapsto \alpha \cdot e^\dagger$ of a complete Elgot algebra. See Example 5.8 for a proof.

4 The Eilenberg-Moore Category of the Monad $\mathbb{R}$

We prove now that the category of all Elgot algebras and solution-preserving morphisms, defined as expected, is the category $\mathcal{A}^{\mathbb{R}}$ of Eilenberg-Moore algebras of the rational monad $\mathbb{R}$ of H, see Remark 2.7.

Throughout this section H denotes a finitary endofunctor of a locally finitely presentable category $\mathcal{A}$. We denote by $\mathcal{A}_{fp}$ a small full subcategory representing all finitely presentable objects of $\mathcal{A}$. Recall the operations $\bullet$ and $\blacksquare$ from Remark 2.13.

Definition 4.1. *Let $(A, \alpha, (-)^\dagger)$, and $(B, \beta, (-)^\ddagger)$ be Elgot algebras. We say that a morphism $h : A \longrightarrow B$ in $\mathcal{A}$* preserves solutions *provided that for every finitary flat equation morphism $e : X \longrightarrow HX + A$ we have the following equation*

$$X \xrightarrow{e^\dagger} A \xrightarrow{h} B \equiv X \xrightarrow{(h \bullet e)^\dagger} B\,. \tag{4.1}$$

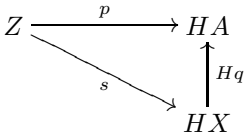
Lemma 4.2. *Every solution-preserving morphism between Elgot algebras is a homomorphism of H-algebras, i.e., we have $h \cdot \alpha = \beta \cdot Hh$.*

Proof. Let $\mathcal{A}_{fp}/A$ be the comma-category of all arrows $q : X \longrightarrow A$ with X in $\mathcal{A}_{fp}$. Since $\mathcal{A}$ is locally finitely presentable, A is a filtered colimit of the canonical diagram $D_A : \mathcal{A}_{fp}/A \longrightarrow \mathcal{A}$ given by $(q : X \longrightarrow A) \longmapsto X$.

Now $\mathcal{A}_{fp}$ is a generator of $\mathcal{A}$, thus, in order to prove the lemma it is sufficient to prove that for every morphism $p : Z \longrightarrow HA$ with Z in $\mathcal{A}_{fp}$ we have

$$h \cdot \alpha \cdot p = \beta \cdot Hh \cdot p. \tag{4.2}$$

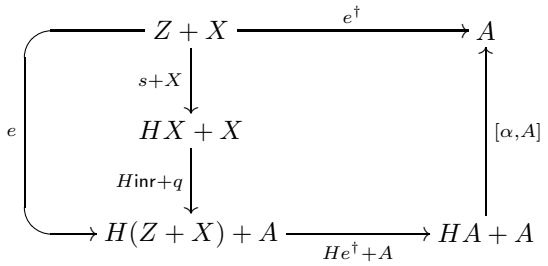
Since H is finitary, it preserves the above colimit D_A. This implies, since $\mathcal{A}(Z, -)$ preserves filtered colimits, that p has a factorization



for some $q : X \longrightarrow A$ in $\mathcal{A}_{fp}/A$ and some s. For the following equation morphism

$$e \equiv Z + X \xrightarrow{s+X} HX + X \xrightarrow{H\mathsf{inr}+q} H(Z + X) + A$$

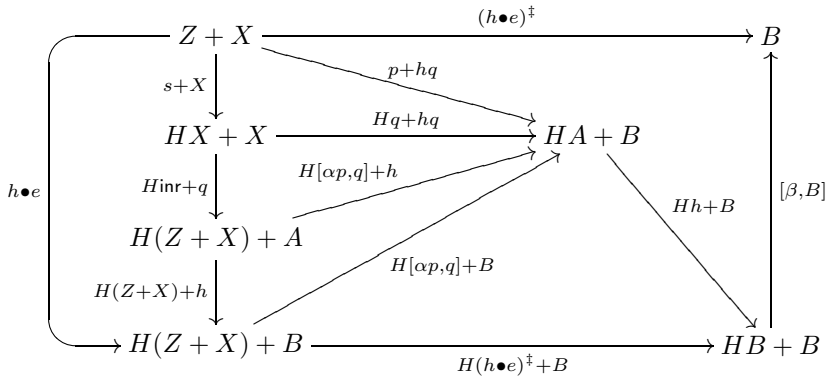
we have a commutative square



Consequently, $e^\dagger \cdot \mathsf{inr} = q$, and this implies $e^\dagger \cdot \mathsf{inl} = \alpha \cdot H(e^\dagger \cdot \mathsf{inr}) \cdot s = \alpha \cdot p$. Since h preserves solutions, we have $h \cdot e^\dagger = (h \bullet e)^\ddagger$ and therefore

$$(h \bullet e)^\ddagger = [h \cdot \alpha \cdot p, h \cdot q]. \tag{4.3}$$

On the other hand, consider the following diagram



It commutes: the outer shape commutes since $(h \bullet e)^\ddagger$ is a solution. For the lower triangle use equation (4.3), and the remaining triangles are trivial. Thus, the upper right-hand part commutes:

$$(h \bullet e)^\ddagger = [\beta \cdot Hh \cdot p, h \cdot q]. \tag{4.4}$$

Now the left-hand components of (4.3) and (4.4) establish the desired equality (4.2).

Example 4.3. The converse of Lemma 4.2 is true for iterative algebras, as proved in [AMV$_2$], but for Elgot algebras in general it is false. In fact, consider the unary algebra $id : A \longrightarrow A$, where $A = \{0, 1\}$. This is an Elgot algebra with the solution structure $(-)^\dagger$ given by the fixed point $0 \in A$, see Example 3.8.

Then $\mathrm{const}_1 : A \longrightarrow A$ is a homomorphism of unary algebras that does not preserve solutions. Indeed, consider the following equation morphism

$$e : \{x\} \longrightarrow \{x\} + A, \quad x \longmapsto x.$$

We have $e^\dagger(x) = 0$, and thus $1 = \mathrm{const}_1 \cdot e^\dagger(x) \neq (\mathrm{const}_1 \bullet e)^\dagger(x) = e^\dagger(x) = 0$.

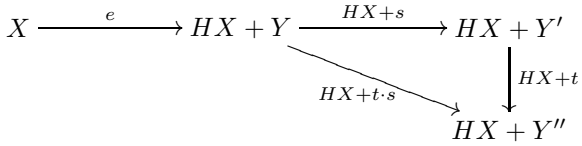
Notation 4.4. We denote by

$$\mathsf{Alg}^\dagger\, H$$

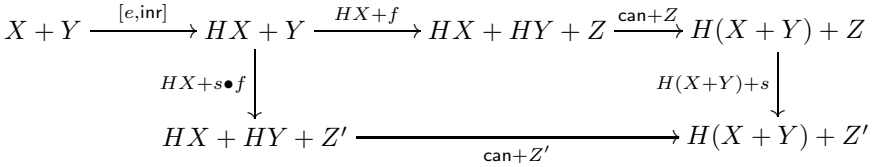
the category of all Elgot algebras and solution-preserving morphisms.

Remark 4.5. For the two operations $\bullet$ and $\blacksquare$ from Remark 2.13 we list some obvious properties that these operations have for all $e : X \longrightarrow HX + Y$, $f : Y \longrightarrow HY + Z$, $s : Z \longrightarrow Z'$ and $t : Z' \longrightarrow Z''$:

1. $id_Y \bullet e = e$. This is trivial.
2. $t \bullet (s \bullet e) = (t \cdot s) \bullet e$.
   See the following diagram



3. $s \bullet (f \blacksquare e) = (s \bullet f) \blacksquare e$.
   See the following diagram



Proposition 4.6. *A free iterative algebra on Y is a free Elgot algebra on Y.*

Proof. (1) We first recall the construction of the free iterative algebra RY on Y presented in [AMV$_2$]. For the functor $H(-) + Y$ denote by EQ_Y the full subcategory of $\mathsf{Coalg}\,(H(-) + Y)$ given by all coalgebras with a finitely presentable carrier, i.e., finitary flat equation morphisms $e : X \longrightarrow HX + Y$. The inclusion functor $\mathrm{Eq}_Y : \mathsf{EQ}_Y \longrightarrow \mathsf{Coalg}\,(H(-) + Y)$ is an essentially small filtered diagram. Put

$$RY = \operatorname{colim} \mathrm{Eq}_Y\,.$$

More precisely, form a colimit of the above diagram Eq_Y. This is a coalgebra RY with the following coalgebra structure

$$i : RY \longrightarrow HRY + Y$$

and with colimit injections

$$e^\sharp : (X, e) \longrightarrow (RY, i) \qquad \text{for all } e : X \longrightarrow HX + Y \text{ in } \mathsf{EQ}_Y.$$

Notice that this colimit is preserved by the forgetful functor $\mathsf{Coalg}\,(H(-) + Y) \longrightarrow \mathcal{A}$ since H is finitary.

The coalgebra structure $i : RY \longrightarrow HRY + Y$ is an isomorphism; its inverse gives an H-algebra structure

$$\rho_Y : HRY \longrightarrow RY$$

and a morphism

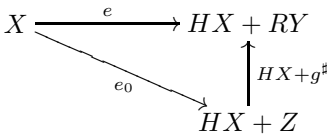
$$\eta_Y : Y \longrightarrow RY.$$

And we proved that the algebra (RY, ρ_Y) is a free iterative H-algebra on Y with the universal arrow η_Y.

Recall further from [AMV$_2$] that the unique solution

$$e^\ddagger : X \longrightarrow RY$$

for every finitary flat equation morphism $e : X \longrightarrow HX + RY$ is obtained as follows. There exists a factorization



(4.5)

with $g : Z \longrightarrow HZ + Y$ in EQ_Y. Define

$$e^{\ddagger} \equiv X \xrightarrow{\mathsf{inl}} X + Z \xrightarrow{(g \blacksquare e_0)^{\sharp}} RY$$

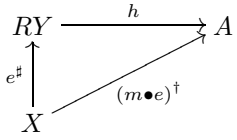
This defines $(-)^{\ddagger}$ from $(-)^{\sharp}$. Conversely, it is trivial to see that the equality

$$e^{\sharp} = (\eta_Y \bullet e)^{\ddagger} \tag{4.6}$$

holds for every $e : X \longrightarrow HX + Y$ in EQ_Y. Finally, the universal arrow η_Y has for any finitely presentable object Y the form $\eta_Y = \mathsf{inr}^{\sharp}$ (for $\mathsf{inr} : Y \longrightarrow HY + Y$).

(2) We are prepared to prove the Proposition. Suppose that $(A, \alpha, (-)^{\dagger})$ is an Elgot algebra and let $m : Y \longrightarrow A$ be a morphism. We are to prove that there exists a unique solution-preserving $h : RY \longrightarrow A$ with $h \cdot \eta_Y = m$.

In order to show the existence, we define a morphism $h : RY \longrightarrow A$ by commutativity of the following triangles



for all $e : X \longrightarrow HX + Y$ in EQ_Y. In fact, h is well-defined, since for any coalgebra homomorphism

$$k : (X, e) \longrightarrow (Z, g)$$

in EQ_Y we have a coalgebra homomorphism

$$k : (X, m \bullet e) \longrightarrow (Z, m \bullet g).$$

Thus,

$$(m \bullet e)^{\dagger} \cdot k = (m \bullet g)^{\dagger}$$

by functoriality, which shows that we obtain a cocone for the diagram Eq_Y. For $e = \mathsf{inr} : Y \longrightarrow HY + Y$, Y finitely presentable, we have $e^{\sharp} = \eta_Y$, thus,

$$\begin{aligned}
h \cdot \eta_Y &= (m \bullet \mathsf{inr})^{\dagger} && \text{(Since } \eta_Y = \mathsf{inr}^{\sharp}) \\
&= [\alpha, A] \cdot (H(m \bullet \mathsf{inr})^{\dagger} + A) \cdot (m \bullet \mathsf{inr}) && \text{(By (2.1))} \\
&= [\alpha, A] \cdot (H(m \bullet \mathsf{inr})^{\dagger} + A) \cdot (HY + m) \cdot \mathsf{inr} && \text{(Definition of } \bullet) \\
&= m\,.
\end{aligned}$$

For arbitrary objects Y the equation $h \cdot \eta_Y = m$ follows easily.

Let us show that h preserves solutions. We have

$$\begin{aligned}
h \cdot e^{\ddagger} &= h \cdot (g \blacksquare e_0)^{\sharp} \cdot \mathsf{inl} && \text{(Definition of } e^{\ddagger}) \\
&= (m \bullet (g \blacksquare e_0))^{\dagger} \cdot \mathsf{inl} && \text{(Definition of } h) \\
&= ((m \bullet g) \blacksquare e_0)^{\dagger} \cdot \mathsf{inl} && \text{(By 4.5(3))} \\
&= ((m \bullet g)^{\dagger} \bullet e_0)^{\dagger} && \text{(compositionality)} \\
&= ((h \cdot g^{\sharp}) \bullet e_0)^{\dagger} && \text{(Definition of } h) \\
&= (h \bullet (g^{\sharp} \bullet e_0))^{\dagger} && \text{(By 4.5(2))} \\
&= (h \bullet e)^{\dagger} && \text{(By (4.5) and the definition of } \bullet)
\end{aligned}$$

Concerning the uniqueness, suppose that h with $h \cdot \eta_Y = m$ preserves solutions, then we have

$$\begin{aligned}
h \cdot e^{\sharp} &= h \cdot (\eta_Y \bullet e)^{\ddagger} && \text{(By (4.6))} \\
&= (h \bullet (\eta_Y \bullet h))^{\dagger} && (h \text{ preserves solutions}) \\
&= ((h \cdot \eta_Y) \bullet e)^{\dagger} && \text{(By 4.5(2))} \\
&= (m \bullet e)^{\dagger}
\end{aligned}$$

which determines h uniquely.

Theorem 4.7. *The category* $\mathsf{Alg}^\dagger H$ *of Elgot algebras is isomorphic to the Eilenberg-Moore category* $\mathcal{A}^{\mathbb{R}}$ *of* $\mathbb{R}$*-algebras for the rational monad* $\mathbb{R}$ *of* H*.*

Remark 4.8. The shortest proof we know is based on Beck's Theorem. But it is not very intuitive. A slightly more technical (and much more illuminating) proof has the following sketch: Denote for any object Y by $(RY, \rho_Y, (-)^\ddagger)$ a free Elgot algebra on Y with a universal arrow $\eta_Y : Y \longrightarrow RY$.

1. For every $\mathbb{R}$-algebra $\alpha_0 : RA \longrightarrow A$ we have an "underlying" H-algebra
$$\alpha \equiv HA \xrightarrow{H\eta_A} HRA \xrightarrow{\rho_A} RA \xrightarrow{\alpha_0} A,$$
and the following formula for solving equations: given a finitary flat equation morphism $e : X \longrightarrow HX + A$ put
$$e^\dagger \equiv X \xrightarrow{(\eta_A \bullet e)^\ddagger} RA \xrightarrow{\alpha_0} A\,.$$
It is not difficult to see that this formula indeed yields a choice of solutions satisfying functoriality and compositionality.
2. Conversely, given an Elgot algebra $\alpha : HA \longrightarrow A$, define $\alpha_0 : RA \longrightarrow A$ as the unique solution-preserving morphism such that $\alpha_0 \cdot \eta_A = id$. It is easy to see that α_0 satisfies the two axioms of an Eilenberg-Moore algebra.
3. It is necessary to prove that the above passages extend to the level of morphisms and they form functors which are inverse to each other.

Proof (Theorem 4.7). By Proposition 4.6 the natural forgetful functor $U : \mathsf{Alg}^\dagger H \longrightarrow \mathcal{A}$ has a left adjoint $Y \longmapsto RY$. Thus, the monad obtained by this adjunction is $\mathbb{R}$. We prove that the comparison functor $K : \mathsf{Alg}^\dagger H \longrightarrow \mathcal{A}^{\mathbb{R}}$ is an isomorphism, using Beck's theorem (see [ML], Theorem 1 in VI.7). Thus, we must prove that U creates coequalizers of U-split pairs. Let $(A, \alpha, (-)^\dagger)$ and $(B, \beta, (-)^\ddagger)$ be Elgot algebras, and $f, g : A \longrightarrow B$ be solution-preserving morphisms with a splitting

$$A \underset{t}{\overset{f}{\underset{g}{\rightrightarrows}}} B \underset{s}{\overset{c}{\rightleftarrows}} C$$

in $\mathcal{A}$ (where $cs = id$, $ft = id$ and $gt = sc$). Since c is, then, an absolute coequalizer of f and g, c is a coequalizer in $\mathsf{Alg}\, H$ for a unique H-algebra structure $\gamma : HC \longrightarrow C$. In fact, the forgetful functor $\mathsf{Alg}\, H \longrightarrow \mathcal{A}$ creates every colimit that H preserves.

It remains to show that C has a unique structure of an Elgot algebra such that

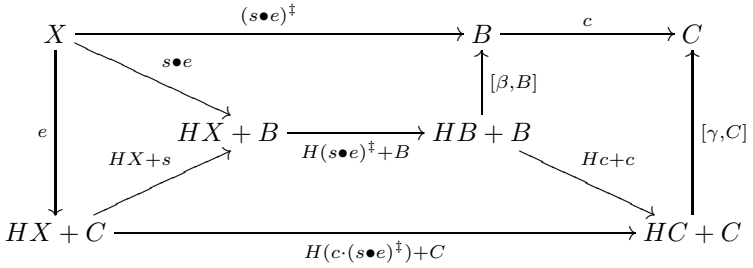
(1) c preserves solutions, and
(2) c is a coequalizer in $\mathsf{Alg}^\dagger H$.

We establish (1) and (2) in several steps.

(a) An Elgot algebra on (C, γ). For every finitary flat equation morphism $e : X \longrightarrow HX + C$ we prove that the following morphism

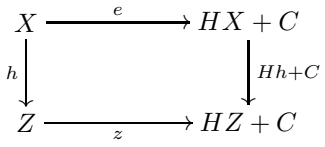
$$e^* \equiv X \xrightarrow{(s \bullet e)^\ddagger} B \xrightarrow{c} C$$

is a solution of e. In fact, the following diagram



clearly commutes.

Functoriality: any coalgebra homomorphism



is, of course, a coalgebra homomorphism

$$h : (X, s \bullet e) \longrightarrow (Z, s \bullet z)\,.$$

Thus,

$$e^* = c \cdot (s \bullet e)^\ddagger = c \cdot (s \bullet z)^\ddagger \cdot h = z^* \cdot h$$

by the functoriality of $(-)^\ddagger$.

Let us prove compositionality: suppose we have finitary flat equation morphisms

$$e : X \longrightarrow HX + Y \quad \text{and} \quad k : Y \longrightarrow HY + C$$

Then we obtain the desired equation as follows:

$$\begin{aligned}
(k^* \bullet e)^* &= c \cdot (s \bullet (k^* \bullet e))^\ddagger && \text{(Definition of } (-)^*) \\
&= c \cdot (s \bullet (c \cdot (s \bullet k)^\ddagger \bullet e))^\ddagger && \text{(Definition of } (-)^*) \\
&= c \cdot ((s \cdot c) \bullet ((s \bullet k)^\ddagger \bullet e))^\ddagger && \text{(see 4.5(2))} \\
&= c \cdot ((g \cdot t) \bullet ((s \bullet k)^\ddagger \bullet e))^\ddagger && (g \cdot t = s \cdot c) \\
&= c \cdot (g \bullet (t \bullet ((s \bullet k)^\ddagger \bullet e)))^\ddagger && \text{(see 4.5(2))} \\
&= c \cdot g \cdot (t \bullet ((s \bullet k)^\ddagger \bullet e))^\dagger && (g \text{ preserves solutions}) \\
&= c \cdot f \cdot (t \bullet ((s \bullet k)^\ddagger \bullet e))^\dagger && (c \cdot f = c \cdot g) \\
&= c \cdot ((f \cdot t) \bullet ((s \bullet k)^\ddagger \bullet e))^\ddagger && (f \text{ preserves solutions and 4.5(2)}) \\
&= c \cdot ((s \bullet k)^\ddagger \bullet e)^\ddagger && (f \cdot t = id \text{ and 4.5(1)}) \\
&= c \cdot ((s \bullet k) \blacksquare e)^\ddagger \cdot \mathsf{inl} && \text{(compositionality for } (-)^\ddagger) \\
&= c \cdot (s \bullet (k \blacksquare e))^\ddagger \cdot \mathsf{inl} && \text{(Since } (s \bullet k) \blacksquare e = s \bullet (k \blacksquare e) \text{ by 4.5(3))} \\
&= (k \blacksquare e)^* \cdot \mathsf{inl} && \text{(Definition of } (-)^*)
\end{aligned}$$

(b) The morphism $c : B \longrightarrow C$ is solution-preserving. In fact, for any finitary flat equation morphism

$$e : X \longrightarrow HX + B$$

we have the desired equation:

$$\begin{aligned}
(c \bullet e)^* &= c \cdot (s \bullet (c \bullet e))^\ddagger && \text{(Definition of } (-)^*) \\
&= c \cdot ((s \cdot c) \bullet e)^\ddagger && \text{(See 4.5(2))} \\
&= c \cdot ((g \cdot t) \bullet e)^\ddagger && (g \cdot t = s \cdot c) \\
&= c \cdot (g \bullet (t \bullet e))^\ddagger && \text{(See 4.5(2))} \\
&= c \cdot g \cdot (t \bullet e)^\dagger && (g \text{ preserves solutions}) \\
&= c \cdot f \cdot (t \bullet e)^\dagger && (c \cdot f = c \cdot g) \\
&= c \cdot (f \bullet (t \bullet e))^\ddagger && (f \text{ preserves solutions}) \\
&= c \cdot ((f \cdot t) \bullet e)^\ddagger && \text{(See 4.5(2))} \\
&= c \cdot (id \bullet e)^\ddagger && (f \cdot t = id) \\
&= c \cdot e^\ddagger && \text{(See 4.5(1))}
\end{aligned}$$

(c) $(-)^*$ is a unique structure of an Elgot algebra such that c is solution-preserving: in fact, for any such solution structure $(-)^*$ and for any finitary flat equation morphism $e : X \longrightarrow HX + B$ we have

$$c \cdot e^\ddagger = (c \bullet e)^* .$$

In particular, this is true for any equation morphism of the form

$$(s \bullet e') \equiv X \xrightarrow{e'} HX + C \xrightarrow{HX+s} HX + B$$

Thus, we conclude

$$\begin{array}{lll} e^* = ((c \cdot s) \bullet e)^* & & (c \cdot s = id \text{ and } 4.5(3)) \\ \quad = (c \bullet (s \bullet e))^* & & (\text{See } 4.5(2)) \\ \quad = c \cdot (s \bullet e)^\ddagger & & (c \text{ preserves solutions}) \end{array}$$

(d) c is a coequalizer of f and g in $\mathsf{Alg}^\dagger H$. In fact, let $h : (B, \beta, (-)^\ddagger) \longrightarrow (D, \delta, (-)^+)$ be a solution-preserving morphism with $h \cdot f = h \cdot g$. There is a unique homomorphism $\overline{h} : C \longrightarrow D$ of H-algebras with $\overline{h} \cdot c = h$ (because c is a coequalizer of f and g in $\mathsf{Alg}\, H$). We prove that $\overline{h}$ is solution-preserving. Let $e : X \longrightarrow HX + C$, be a finitary flat equation morphism. Then we have

$$\begin{array}{lll} \overline{h} \cdot e^* = \overline{h} \cdot c \cdot (s \bullet e)^\ddagger & & (\text{Definition of } (-)^*) \\ \quad = h \cdot (s \bullet e)^\ddagger & & (h = \overline{h} \cdot c) \\ \quad = (h \bullet (s \bullet e))^+ & & (h \text{ preserves solutions}) \\ \quad = ((h \cdot s) \bullet e)^+ & & (\text{See } 4.5(2)) \\ \quad = ((\overline{h} \cdot c \cdot s) \bullet e)^+ & & (h = \overline{h} \cdot c) \\ \quad = (\overline{h} \bullet e)^+ & & (c \cdot s = id) \end{array}$$

as desired. This completes the proof.

Example 4.9. Let A be a join semilattice. Recall from Example 3.2 the function $\alpha : RA \longrightarrow A$ assigning to a rational binary tree t in RA the join of the labels of all leaves of t in A. Since joins commute with joins it follows that this is the structure of an Eilenberg-Moore algebra on A. Thus, A is an Elgot algebra as described in Example 3.2.

5 Complete Elgot Algebras

Recall our standing assumptions that H is an endofunctor of a category $\mathcal{A}$ with finite coproducts. Stefan Milius has established in [M] that for every object-mapping T of $\mathcal{A}$ the following three statements are equivalent:

(a) for every object Y, TY is a final coalgebra of $H(-) + Y$
(b) for every object Y, TY is a free completely iterative H-algebra on Y, and
(c) T is the functor part of a free completely iterative monad $\mathbb{T}$ on H.

See also [AAMV] where the monad $\mathbb{T}$ is described and the implication that (a) implies (c) is proved.

We are going to add another equivalent item to the above list, bringing complete Elgot algebras into the picture. The statements (a) to (c) are equivalent to

(d) for every object Y, TY is a free complete Elgot algebra on Y.

Furthermore, recall from [AAMV] that H is *iteratable* if there exist objects TY such that one of the above equivalent statements holds. We will describe for every iteratable endofunctor the category $\mathcal{A}^\mathbb{T}$ of Eilenberg–Moore algebras—it is isomorphic to the category of complete Elgot algebras of H.

Example 5.1. For a polynomial endofunctor H_Σ of Set the above monad is the monad T_Σ of all (finite and infinite) Σ-trees.

In the following result the concept of *solution-preserving morphism* is defined for complete Elgot algebras analogously to Definition 4.1: the equation (4.1) holds for *all* flat equation morphisms e. We denote by

$$\mathsf{Alg}_c^\dagger\, H$$

the category of all complete Elgot algebras and solution-preserving morphisms.

Lemma 5.2. *Every solution-preserving morphism between complete Elgot algebras is a homomorphism of H-algebras.*

Proof. Let $(A, \alpha, (-)^\dagger)$ and $(B, \beta, (-)^\ddagger)$ be complete Elgot algebras. Suppose that $h : A \longrightarrow B$ is a solution-preserving morphism, and consider the equation morphism

$$e \equiv HA + A \xrightarrow{H\mathsf{inr}+A} H(HA + A) + A$$

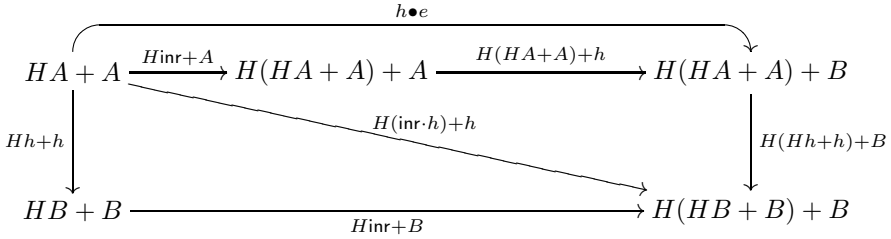
Its solution fulfils $e^\dagger = [\alpha, A] : HA + A \longrightarrow A$. In fact, the following diagram

$$\begin{array}{ccc} HA + A & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle H\mathsf{inr}+A}\downarrow & & \uparrow{\scriptstyle [\alpha,A]} \\ H(HA + A) + A & \xrightarrow[He^\dagger+A]{} & HA + A \end{array}$$

commutes. Thus, $e^\dagger \cdot \mathsf{inr} = id$, and then it follows that $e^\dagger \cdot \mathsf{inl} = \alpha$. Since h preserves solutions we know that $h \cdot \alpha$ is the left-hand component of the solution of the following equation morphism

$$h \bullet e \equiv HA + A \xrightarrow{H\mathsf{inr}+A} H(HA + A) + A \xrightarrow{H(HA+A)+h} H(HA + A) + B\,,$$

i. e., $(h \bullet e)^\ddagger \cdot \mathsf{inl} = h \cdot \alpha$. Now consider the diagram



which trivially commutes. Hence, $Hh + h$ is a morphism of equations from $h \bullet e$ to $H\mathsf{inr} + B$. By a similar argument as for $e^\dagger$ above we obtain $[\beta, B] = (H\mathsf{inr} + B)^\ddagger$. Thus, by functoriality we conclude that

$$h \cdot \alpha = (h \bullet e)^\ddagger \cdot \mathsf{inl} = [\beta, B] \cdot (Hh + h) \cdot \mathsf{inl} = \beta \cdot Hh\,,$$

i. e., h is an H-algebra homomorphism.

Theorem 5.3. *Let Y be an object of $\mathcal{A}$. Then the following are equivalent:*

(1) TY is a final coalgebra of $H(-) + Y$, and
(2) TY is a free complete Elgot algebra on Y.

Before proving this theorem, we need a technical lemma:

Construction 5.4. Let $(A, \alpha, (-)^\dagger)$ be a complete Elgot algebra. For every morphism $m : Y \longrightarrow A$ we construct a new complete Elgot algebra on $HA + Y$ as follows:

1. The algebra structure is

$$H(HA+Y) \xrightarrow{H[\alpha,m]} HA \xrightarrow{\mathsf{inl}} HA+Y\,.$$

2. The choice $(-)^\ddagger$ of solutions is as follows: for every flat equation morphism $e : X \longrightarrow HX + HA + Y$ consider the flat equation morphism

$$\overline{e} \equiv X \xrightarrow{e} HX + HA + Y \xrightarrow{HX+[\alpha,m]} HX + A\,,$$

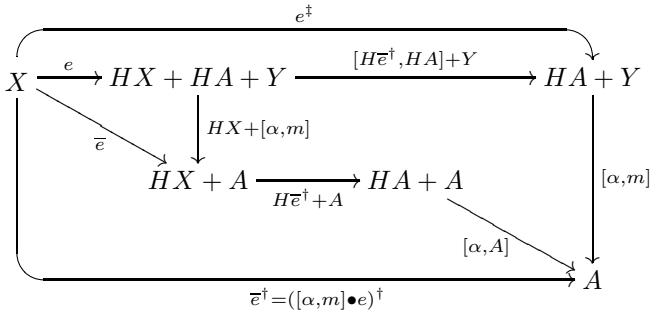
and put

$$e^\ddagger \equiv X \xrightarrow{e} HX + HA + Y \xrightarrow{[H\overline{e}^\dagger, HA]+Y} HA + Y\,.$$

Notice that $\overline{e} = [\alpha, m] \bullet e$.

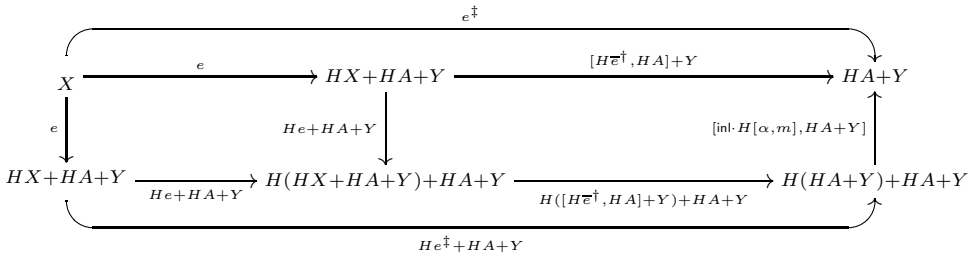
Lemma 5.5. *The above construction defines a complete Elgot algebra such that* $[\alpha, m] : HA + Y \longrightarrow A$ *is a solution-preserving morphism into the original algebra.*

Proof. (1) The morphism $[\alpha, m]$ is solution-preserving: In fact, for any flat equation morphism $e : X \longrightarrow HX + HA + Y$ we have the following commutative diagram

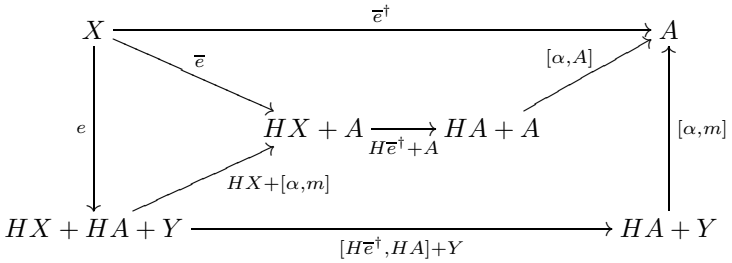


The lower left-hand part commutes since $\overline{e}^\dagger$ solves $\overline{e}$; the upper part is the definition of $(-)^\ddagger$, the left-hand triangle is the definition of $\overline{e}$, and all components of the inner right-hand part are clear.

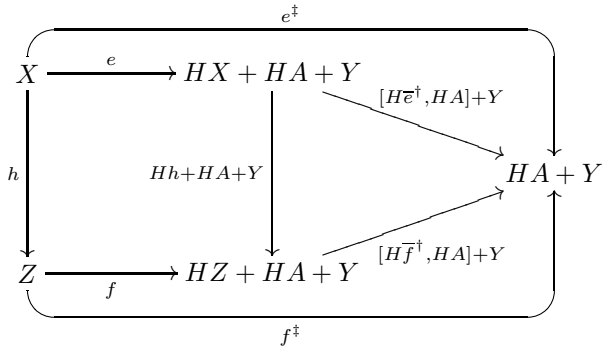
(2) The morphism $e^\ddagger$ is a solution of e. In fact, the following diagram



commutes: the upper and lower part as well as the left-hand square are obvious, and so are the middle and right-hand components of the right-hand square. To see that the left-hand component commutes, we remove H and observe that the following diagram commutes:



(3) Functoriality: Suppose we have a morphism $h : e \longrightarrow f$ of equations. Then $h : \overline{e} \longrightarrow \overline{f}$ is also one, und we obtain the following diagram

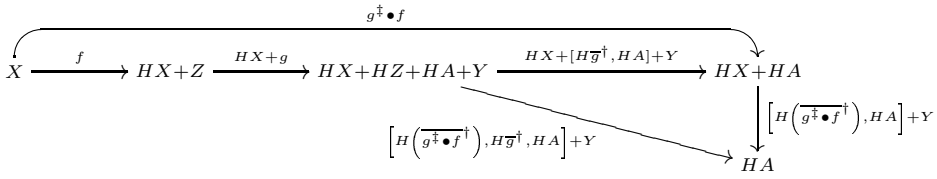


It commutes: in the triangle the components with domains HA and Y are clear, for the left-hand component remove H and use the functoriality of $(-)^\dagger$, and all other parts are obvious.

(4) Compositionality: Suppose we have two equation morphisms

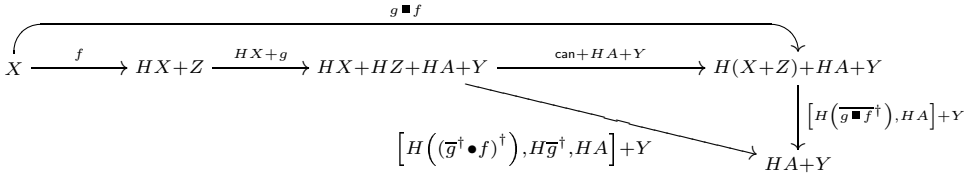
$$f : X \longrightarrow HX + Y \quad \text{and} \quad g : Z \longrightarrow HZ + HA + Y\,.$$

Observe that $(g^\ddagger \bullet f)^\ddagger$ is the following morphism



(5.1)

and $(g \blacksquare f)^\ddagger \cdot \mathsf{inl}$ is the following morphism



(5.2)

In fact, to see that the last triangle commutes consider the components separately. The right-hand one with domain $HA + Y$ is trivial, and for the left-hand one with domain $HX + HZ$ it suffices to observe the following equations:

$$\begin{aligned}
\overline{g \blacksquare f}^\dagger &= ([\alpha, m] \bullet (g \blacksquare f))^\dagger && \text{(Definition of } \overline{g \blacksquare f}\text{)} \\
&= (([\alpha, m] \bullet g) \blacksquare f)^\dagger && \text{(By 4.5(3))} \\
&= \left[\left(([\alpha, m] \bullet g)^\dagger \bullet f\right)^\dagger, ([\alpha, m], \bullet g)^\dagger\right] && \text{(See (2.7))} \\
&= \left[(\overline{g}^\dagger \bullet f)^\dagger, \overline{g}^\dagger\right] && \text{(Definition of } \overline{g}\text{)}
\end{aligned}$$

To show the desired identity of the morphisms in (5.1) and (5.2) it suffices to prove that the slanting arrows in those diagrams are equal. The last three components are clear, and for the first one the following equations are sufficient:

$$\begin{aligned}
\overline{g}^\dagger \bullet f &= ([\alpha, m] \bullet g)^\dagger \bullet f && \text{(Definition of } \overline{g}\text{)} \\
&= ([\alpha, m] \cdot g^\ddagger) \bullet f && \text{(See part (1) of the proof)} \\
&= [\alpha, m] \bullet (g^\ddagger \bullet f) && \text{(By 4.5(2))} \\
&= \overline{g^\ddagger \bullet f} && \text{(Definition of } \overline{g^\ddagger \bullet f}\text{)}
\end{aligned}$$

This completes the proof.

Proof (Theorem 5.3). Statement (1) is equivalent to

(1') TY is a free CIA on Y,

see Theorems 2.8 and 2.10 of [M]. We prove now that (2) is equivalent to (1). We first observe that for a free Elgot algebra on Y, $(TY, \tau_Y, (-)^\dagger)$, with a universal arrow $\eta_Y : Y \longrightarrow TY$, the morphism $[\tau_Y, \eta_Y] : HTY + Y \longrightarrow TY$ is an isomorphism. In fact, by Lemma 5.5, $HTY + Y$ carries the structure of a complete Elgot algebra and $j = [\tau_Y, \eta_Y]$ is solution-preserving and fulfils $j \cdot \mathsf{inr} = \eta_Y$. Invoke the freeness of TY to obtain a unique solution-preserving morphism $i : TY \longrightarrow HTY + Y$ such that $i \cdot \eta_Y = \mathsf{inr}$. It follows that $j \cdot i = id$. By Lemma 5.2, i is an H-algebra homomorphism. Thus the following square

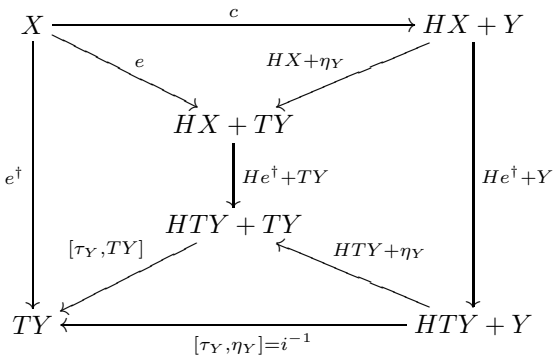
$$\begin{array}{ccc} HTY + Y & \xrightarrow{\;j\;} & TY \\ {\scriptstyle Hi+Y}\downarrow & & \downarrow{\scriptstyle i} \\ H(HTY + Y) + Y & \xrightarrow[Hj+Y]{} & HTY + Y \end{array}$$

commutes, whence $i \cdot j = id$.

Proof of (2) $\Rightarrow$ (1). Let $(TY, \tau_Y, (-)^\dagger)$ be a free complete Elgot algebra on Y with a universal arrow $\eta_Y : Y \longrightarrow TY$. Then $[\tau_Y, \eta_Y] : HTY + Y \longrightarrow TY$ is an isomorphism with an inverse i. We prove that (TY, i) is a final coalgebra of $H(-) + Y$. So let $c : X \longrightarrow HX + Y$ be any coalgebra, and form the flat equation morphism

$$e \equiv X \xrightarrow{\;c\;} HX + Y \xrightarrow{\;HX+\eta_Y\;} HX + TY\,. \tag{5.3}$$

Then $e^\dagger$ is a coalgebra homomorphism from (X, c) to (TY, i); in fact, it suffices to establish that the diagram

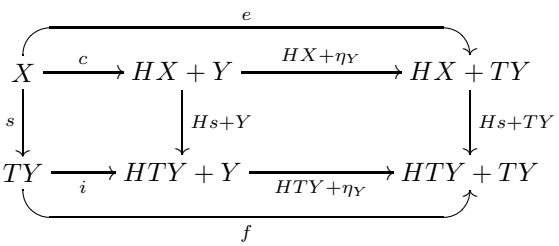


commutes. The upper part is (5.3), the left-hand part commutes since $e^\dagger$ is a solution of e, the right-hand one commutes by the naturality of η, and the lower part is obvious.

Now suppose that s is a coalgebra homomorphism from (X, c) to (TY, i). We prove that $s = e^\dagger$. Observe first that s is a morphism of equations from e to the following flat equation morphism

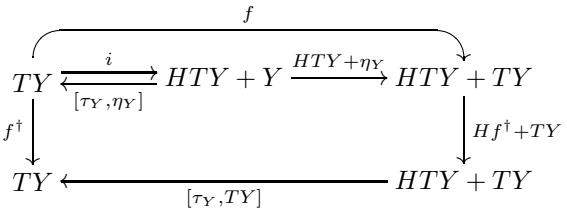
$$f \equiv TY \xrightarrow{\;i\;} HTY + Y \xrightarrow{\;HTY+\eta_Y\;} HTY + TY\,, \tag{5.4}$$

In fact, the following diagram



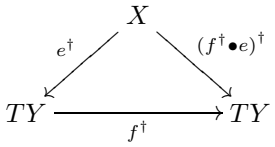
commutes: the left-hand square does since s is a coalgebra homomorphism, the right-hand one by the naturality of η and the upper and lower parts are due to (5.3) and (5.4). By functoriality of $(-)^\dagger$ we obtain $f^\dagger \cdot s = e^\dagger$. We shall show below that $f^\dagger : TY \longrightarrow TY$ is a solution-preserving map with $f^\dagger \cdot \eta_Y = \eta_Y$. By the freeness of TY, we then conclude that $f^\dagger = id$, whence $e^\dagger = s$ as desired.

To see that $f^\dagger \cdot \eta_Y = \eta_Y$ consider the following diagram



which commutes since $f^\dagger$ is a solution of f. Follow the right-hand component of the coproduct $HTY + Y$ to see the desired equation.

Now to complete our proof we must show that the following triangle

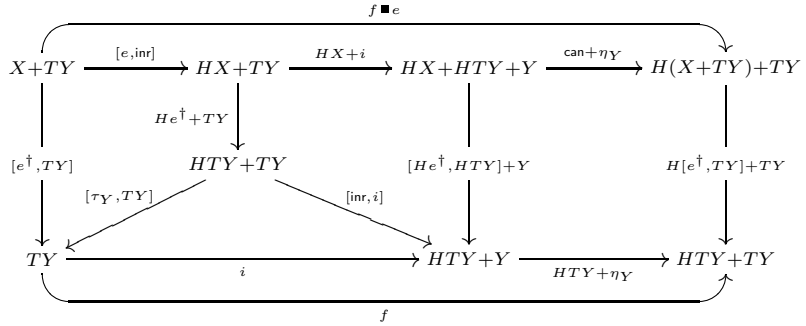


(5.5)

commutes for any equation morphism $e : X \longrightarrow HX + TY$. Notice first that

$$f^\dagger \bullet e^\dagger = (f \blacksquare e)^\dagger \cdot \mathsf{inl} : X \longrightarrow TY \tag{5.6}$$

by compositionality. Furthermore, we have an equation morphism $[e^\dagger, TY] : f \blacksquare e \longrightarrow f$ since the following diagram



(5.7)

commutes. By functoriality we obtain the following equality

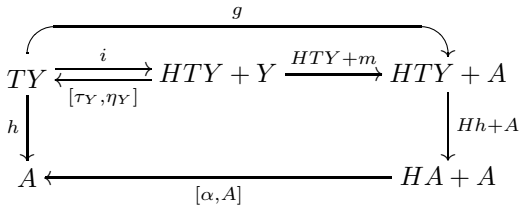
$$f^\dagger \cdot [e^\dagger, TY] = (f \blacksquare e)^\dagger ,$$

whose left-hand component proves due to (5.6) the desired commutativity of (5.5).

(1') $\Rightarrow$ (2). We only need to show the universal property. Suppose that $(TY, \tau_Y, (-)^\dagger)$ is a free CIA on Y with a universal arrow $\eta_Y : Y \longrightarrow TY$. Due to the equivalence of (1) and (2), $[\tau_Y, \eta_Y]$ has an inverse i, and (TY, i) is a final coalgebra of the functor $H(-) + Y$. Now let $(A, \alpha, (-)^\ddagger)$ be a complete Elgot algebra and let $m : Y \longrightarrow A$ be a morphism of $\mathcal{A}$. Solve the following equation morphism

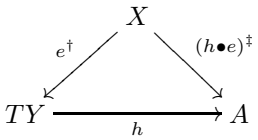
$$g \equiv \; TY \xrightarrow{\;i\;} HTY + Y \xrightarrow{\;HTY+m\;} HTY + A$$

in A to obtain a morphism $h = g^\ddagger : TY \longrightarrow A$. We first check that $h \cdot \eta_Y = m$. In fact, the following diagram



commutes since h is a solution of g. Consider the right-hand component of the coproduct $HTY+Y$ to obtain the desired equation.

Next let us show that h is a solution-preserving morphism. That is, we show that for any equation morphism $e : X \longrightarrow HX + TY$ the triangle

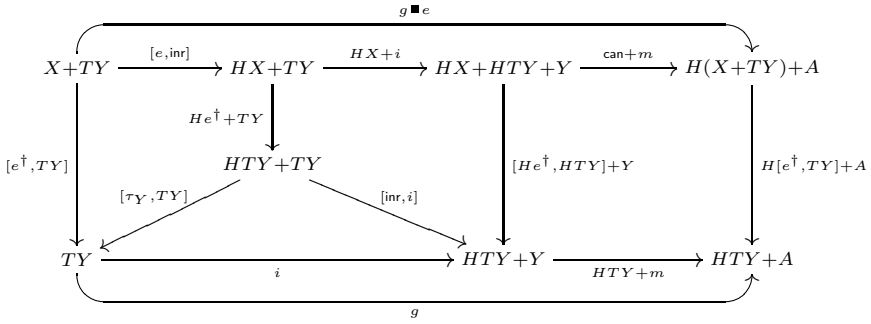


(5.8)

commutes. Since $h = g^\ddagger$,

$$(h \bullet e)^\ddagger = (g \blacksquare e)^\ddagger \cdot \mathsf{inl} : X \longrightarrow A \tag{5.9}$$

due to compositionality of $(-)^\ddagger$. Moreover, $[e^\dagger, TY]$ is an equation morphism from $g \blacksquare e$ to g. In fact, consider the following commutative diagram



which is analogous to Diagram (5.7). By the functoriality of $(-)^\ddagger$ we obtain the equation

$$g^\ddagger \cdot [e^\dagger, TY] = (g \blacksquare e)^\ddagger$$

whose left-hand component is due to (5.9) the desired (5.8). Thus, h is solution-preserving.

To show uniqueness suppose that $h : TY \longrightarrow A$ is any solution-preserving morphism with $h \cdot \eta_Y = m$. Observe that we have $g = h \bullet f$, where f is the equation morphism of (5.4). Since h preserves solutions we have

$$g^\ddagger = (h \bullet f)^\ddagger = h \cdot f^\dagger \,.$$

To complete the proof it suffices to show that $f^\dagger = id$. This can be done with precisely the same argument as in the first part of the proof of Theorem 5.3. One shows that $f^\dagger : TY \longrightarrow TY$ is a solution-preserving morphism such that $f^\dagger \cdot \eta_Y = \eta_Y$. From the universal property of the free CIA TY it follows that $f^\dagger = id$, see also Proposition 2.3 in [M].

Corollary 5.6. *For any endofunctor $H : \mathcal{A} \longrightarrow \mathcal{A}$ the following are equivalent:*

(1) H is iteratable, i. e., there exist final coalgebras of all functors $H(-) + Y$
(2) there exist free completely iterative H-algebras on every object Y
(3) there exist free complete Elgot algebras on every object Y.

Proof. See [M], Corollary 2.11 for (1) $\Leftrightarrow$ (2). The equivalence (2) $\Leftrightarrow$ (3) follows from Theorem 5.3.

Theorem 5.7. *If H is an iteratable functor, then the category $\mathsf{Alg}_c^\dagger H$ of complete Elgot algebras is isomorphic to the Eilenberg–Moore category $\mathcal{A}^{\mathbb{T}}$ of monadic $\mathbb{T}$-algebras (for the free completely iterative monad $\mathbb{T}$ of H).*

Proof. By Corollary 5.6, the natural forgetful functor $U : \mathsf{Alg}_c^\dagger H \longrightarrow \mathcal{A}$ has a left adjoint $Y \longmapsto TY$. Thus, the monad obtained by this adjunction is $\mathbb{T}$. To prove that the comparison functor $K : \mathsf{Alg}_c^\dagger H \longrightarrow \mathcal{A}^{\mathbb{T}}$ is an isomorphism use Beck's Theorem. In fact, the argument that U creates coequalizers of U-split pairs is entirely analogous to that of Theorem 4.7.

Example 5.8. Let A be a complete lattice. Recall from Example 3.10 the function $\alpha : TA \longrightarrow A$ assigning to every binary tree t in TA the join of all labels of leaves of t in A. Since joins commute with joins it follows that $\alpha : TA \longrightarrow A$ is the structure of an Eilenberg-Moore algebra on A. Thus, A is a complete Elgot algebra as described in Example 3.10.

6 Summary and Future Work

The concept of Elgot algebra introduced in our paper formalizes algebras in which finitary flat equation morphisms have solutions satisfying two simple axioms: one for change of parameters and one for simultaneous recursion. And, analogously, complete Elgot algebras are algebras in which flat equation morphisms (not necessarily finitary) have solutions subject to the same two axioms. Such algebras can be used for interpreted semantics of recursive program schemes such as (1.1). In view of the simplicity of the two axioms we consider this is a success. Moreover, the structure of Elgot algebras is provided canonically by Elgot's iterative theories: Elgot algebras are the monadic algebras of the free iterative theory (as described by Calvin Elgot et al. for signatures in [EBT] and by the authors in [AMV$_1$,AMV$_2$] for general endofunctors). And complete Elgot algebras are the monadic algebras of the free completely iterative monad of Calvin Elgot et al. [EBT] (generalized by Stefan Milius in [M]).

For the important "in-between" variant of algebraic trees of Bruno Courcelle [C], i. e., precisely all trees obtained by tree unfoldings of recursive program schemes, no abstract treatment has been presented so far. The present authors are planning to work in a setting in which abstract algebraic trees can be treated. The basic category is, however, not Set, but $\mathsf{Fin}(\mathsf{Set})$, the category of all finitary endofunctors of Set. This category is locally finitely presentable, and that was one reason for presenting our theory in such general categories, not only in Set.

The function $e \longmapsto e^\dagger$ which is part of an Elgot algebra extends canonically from the above flat equation morphisms e to a much broader class of "rational" equation morphisms—another topic of our planned future research. In that sense one gets close to iteration algebras of Zoltan Ésik [É]. The relationship of the latter to Elgot algebras needs further investigation.

Finally, this paper can be considered as part of a program proposed by Lawrence Moss to rework the theory of recursive program schemes and their semantics using coalgebraic methods. We believe that our paper contributed by presenting a "suitable" notion of algebra of a functor which can be used for interpreted semantics or recursive program schemes. We do not have the space to treat this semantics in our paper. This is the topic of the forthcoming paper [MM], where basic results of a categorical theory of recursive program schemes are presented. In that paper the authors introduce a general notion of recursive program scheme (rps), and they prove that any guarded rps has a unique "uninterpreted" solution in the final coalgebra of the functor describing the given operations. Furthermore, it is proved that an interpreted solution can be given to a recursive program scheme in any complete Elgot algebra, and that this solution is unique in case of a CIA. Finally, the fundamental result that every interpreted solution factors through an uninterpreted one is proved. As applications one obtains the classical theory using continuous algebras or completely metrizable ones as interpretations. New applications include, for example, recursively defined operations satisfying extra conditions like commutativity, or applications in non-well founded sets or measure spaces.

We admit that the whole program is at this point still at a beginning phase and so far has not yet produced many new results in semantics that go beyond what can be done with the well-established classical methods. However, we strongly believe that our approach deepens the understanding of the mechanisms at work in algebraic semantics, with categorical results of great conceptual clarity. We hope that this will eventually lead to new insights and results for the semantics of recursive computations.

References

[AAMV] P. Aczel, J. Adámek, S. Milius and J. Velebil, Infinite Trees and Completely Iterative Theories: A Coalgebraic View, *Theoret. Comput. Sci.* 300 (2003), 1–45.

[AMV$_1$] J. Adámek, S. Milius and J. Velebil, Free iterative theories: a coalgebraic view, *Math. Structures Comput. Sci.* 13 (2003), 259–320.

[AMV$_2$] J. Adámek, S. Milius and J. Velebil, From Iterative Algebras to Iterative Theories, *Electron. Notes Theor. Comput. Sci.* 106 (2004), 3–24, full version is available from `http://www.iti.cs.tu-bs.de/~milius/`.

[AMV$_3$] J. Adámek, S. Milius and J. Velebil, Elgot Algebras (Extended Abstract), to appear in *Electron. Notes Theor. Comput. Sci.*

[AR] J. Adámek and J. Rosický, *Locally presentable and accessible categories*, Cambridge University Press, 1994.

[ARu] P. America and J. Rutten, Solving Reflexive Domain Equations in a Category of Complete Metric Spaces, in: *Proc. 3rd Workshop on the Mathematical Foundations of Programming Language Semantics*, M. Main, A. Melton, M. Mislove and D. Schmidt, eds., Springer-Verlag, LNCS 298, 254–288.

[BÉ] S. L. Bloom and Z. Ésik, *Iteration Theories: The Equational Logic of Iterative Processes*, EATCS Monograph Series on Theoretical Computer Science, Springer-Verlag, 1993.

[C] B. Courcelle, Fundamental properties of infinite trees, *Theoret. Comput. Sci.* 25 (1983), no. 2, 95–169.

[Ei] S. Eilenberg, The category $\mathcal{C}$. Unpublished manuscript.

[El] C. C. Elgot, Monadic Computation and Iterative Algebraic Theories, in: *Logic Colloquium '73* (eds: H. E. Rose and J. C. Shepherdson), North-Holland Publishers, Amsterdam, 1975.

[EBT] C. C. Elgot, S. L. Bloom and R. Tindell, On the Algebraic Structure of Rooted Trees, *J. Comput. System Sci.* 16 (1978), 361–399.

[É] Z. Ésik, Algebras of iteration theories, *J. Comput. System Sci.* 27 (1983), 291–303.

[G] S. Ginali, Regular trees and the free iterative theory, *J. Comput. System Sci.* 18 (1979), 228–242.

[GTWW] J. Goguen, J. Thatcher, E. Wagner and J. Wright, Initial algebra semantics and continous algebras, *J. ACM* 24 (1977), 68–95.

[Ha] M. Hasegawa, Recursion from Cyclic Sharing: Traced Monoidal Categories and Models of Cyclic Lambda Calculi, *Proc. 3rd International Conference on Typed Lambda Calculi and Applications*, Springer LNCS 1210, 196–213, 1997.

[JSV] A. Joyal, R. Street and D. Verity, Traced Monoidal Categories, *Math. Proc. Cambridge Philos. Soc.* 119 (3), 447–468.

[ML] S. MacLane, *Categories for the Working Mathematician*, Springer-Verlag, 2$^{\text{nd}}$ edition, 1998.

[M] S. Milius, Completely Iterative Algebras and Completely Iterative Monads, accepted for publication in *Inform. and Comput.*

[MM] S. Milius and L. Moss, The Category Theoretic Solution of Recursive Program Schemes, extended abstract to appear in Lecture Notes in Comput. Sci., full version available from `http://www.iti.cs.tu-bs.de/~milius/`

[Mo] L. S. Moss, Recursion and corecursion have the same equational logic, *Theoret. Comput. Sci.* 294 (2003), 233–267.

[N] E. Nelson, Iterative Algebras, *Theoret. Comput. Science* 25 (1983), 67–94.

[P] G. Plotkin, *Domains*, The "Pisa" notes, available from `http://www.dcs.ed.ac.uk/home/gdp/`, 1983.

[SP$_1$] A. Simpson and G. Plotkin, Complete Axioms for Categorical Fixed-point Operators, Proceedings of the Fifteenth Annual IEEE Symposium on Logic in Computer Science 2000, 30–41.

[SP$_2$] M. B. Smyth and G. Plotkin, The Category-Theoretic Solution of Recursive Domain Equations, *SIAM J. Comp.* 11 (4), 1982, 761–783.

[T] J. Tiuryn, Unique Fixed Points vs. Least Fixed Points, *Theoret. Comput. Sci.* 12 (1980), 229–254.

Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®
Theoretical Computer Science 300 (2003) 1–45
Theoretical Computer Science
www.elsevier.com/locate/tcs

Fundamental Study

Infinite trees and completely iterative theories: a coalgebraic view

Peter Aczel[a], Jiří Adámek[b,*,1], Stefan Milius[b], Jiří Velebil[b,1]

[a] *Department of Mathematics and Computer Science, Manchester University, UK*
[b] *Institute of Theoretical Computer Science, Technical University, Braunschweig, Germany*

Received 7 December 2001; received in revised form 29 July 2002; accepted 12 September 2002
Communicated by Z. Esik

Abstract

Infinite trees form a free completely iterative theory over any given signature—this fact, proved by Elgot, Bloom and Tindell, turns out to be a special case of a much more general categorical result exhibited in the present paper. We prove that whenever an endofunctor H of a category has final coalgebras for all functors $H(\_)+X$, then those coalgebras, TX, form a monad. This monad is completely iterative, i.e., every guarded system of recursive equations has a unique solution. And it is a free completely iterative monad on H. The special case of polynomial endofunctors of the category Set is the above mentioned theory, or monad, of infinite trees.

This procedure can be generalized to monoidal categories satisfying a mild side condition: if, for an object H, the endofunctor $H\otimes\_+I$ has a final coalgebra, T, then T is a monoid. This specializes to the above case for the monoidal category of all endofunctors.

Keywords: Completely iterative theory; Monad; Coalgebra; Solution Theorem; Monoidal category

1. Introduction

Our paper presents an application of corecursion, i.e., of the construction method using final coalgebras, to the theory of iterative equation systems. Recall that equations

* Corresponding author. Tel.: +49-531-3919521; fax: +49-531-3919529.
E-mail addresses: petera@cs.man.ac.uk (P. Aczel), adamek@iti.cs.tu-bs.de (J. Adámek), milius@iti.cs.tu-bs.de (S. Milius), velebil@iti.cs.tu-bs.de (J. Velebil).
[1] The support of the Grant Agency of the Czech Republic under the Grant No. 201/99/0310 is gratefully acknowledged.

0304-3975/03/$ - see front matter
PII: S0304-3975(02)00728-4

such as

$$\begin{aligned} x_1 &\approx x_2 \diamond a \\ x_2 &\approx x_1 \diamond b \end{aligned} \tag{1.1}$$

have unique solutions in the realm of infinite expressions. In our case, the solution is $x_1{}^\dagger = (((\ldots \diamond b) \diamond a) \diamond b) \diamond a$ and $x_2{}^\dagger = (((\ldots \diamond a) \diamond b) \diamond a) \diamond b$. Such infinite expressions, or infinite trees, have been studied in the 1970s in connection with (potentially infinite) computations, where various additional structures were introduced with the aim of formalizing an infinite computation as a join of finite approximations in a CPO, see e.g. [18], or as a limit of a Cauchy sequence of approximations in a complete metric space, see e.g. [10]. A different approach, not using additional structures such as ordering or metric, has been taken by Elgot and his co-authors, see, e.g. [15,16]. The above system (1.1) is an example of a system of *iterative equations* using a set $X = \{x_1, x_2\}$ of variables and a set $Y = \{a, b\}$ of parameters. Given a signature Σ (here consisting of a single binary symbol $\diamond$) a system of iterative equations consists of equations

$$x \approx e(x) \quad (\text{one for every variable } x \text{ in } X)$$

whose right-hand sides are finite or infinite Σ-labelled trees $e(x)$ over the set $X + Y$. That is, trees with leaves labelled by variables, parameters or nullary symbols, and internal nodes with n children labelled by n-ary symbols. The symbol $\approx$ indicates a formal equation, whereas $=$ means the identity of the two sides. A *solution* of the system of equations is a collection

$$e^\dagger(x) \quad (x \in X)$$

of Σ-labelled trees over Y, i.e., trees without variables, such that the substitution of $e^\dagger(x)$ for x, for all variables x, turns the formal equations into identities. That is, for every $x_0 \in X$ we have

$$e^\dagger(x_0) = e(x_0)[e^\dagger(x)/x].$$

The given system is called *guarded* provided that none of the right-hand sides is a single variable. Every guarded system has a unique solution.

In the present paper we show that a coalgebraic approach makes it possible to study solutions of iterative equations without any additional (always a bit arbitrary) structure—that is, we can simply work in Set, the category of sets. We use the simple and well-known fact that for polynomial endofunctors H of Set the algebra of all (finite and infinite) properly labelled trees is a final H-coalgebra. Well, this is not enough: what we need is working with "trees with variables", i.e., given a set X of variables, we work with trees whose internal nodes are labelled by operations, and leaves are labelled by variables and constants. This is a final coalgebra again: not for the original functor, but for the functor

$$H(\_) + X : \mathsf{Set} \to \mathsf{Set}$$

We are going to show that for every polynomial functor $H : \mathsf{Set} \to \mathsf{Set}$

(a) final coalgebras TX of the functors $H(\_)+X$ form a monad, called the *completely iterative monad generated by* H,

(b) there is also a canonical structure of an H-algebra on each TX, and all these canonical H-algebras form the Kleisli category of the completely iterative monad, and

(c) the H-algebra TX has unique solutions of all guarded systems of iterative equations.

A surprising feature of the result we prove is its generality: this has nothing to do with polynomiality of H, nor with the base category Set. In fact, given an endofunctor H of any category $\mathscr{A}$ with binary coproducts, and assuming that each $H(\_)+X$ has a final coalgebra (such functors are called *iteratable*) then (a)–(c) hold.

The above system (1.1) corresponds to the polynomial functor expressing one binary operation, $\diamond$, i.e., to the functor $HZ = Z \times Z$. A final coalgebra TX of $Z \mapsto Z \times Z + X$ can be described as the coalgebra of all finite and infinite binary trees with leaves labelled in X. System (1.1) describes a function from $X = \{x_1, x_2\}$ to the set $T(X+Y)$ of trees over variables from X and parameters from $Y = \{a, b\}$. Here we have

$$e : X \longrightarrow T(X+Y), \quad x_1 \mapsto \begin{array}{ccc} & \diamond & \\ x_2 & & a \end{array}, \quad x_2 \mapsto \begin{array}{ccc} & \diamond & \\ x_1 & & b \end{array}$$

The above concept of solution is categorically expressed by a morphism

$$e^\dagger : X \to TY$$

characterized by the property that $e^\dagger$ is equal to the composite of $e : X \to T(X+Y)$ and the substitution morphism $T(X+Y) \to TY$ leaving parameters intact and substituting $e^\dagger(x)$ for $x \in X$. This substitution is given by the function $s = [e^\dagger, \eta_Y] : X + Y \to TY$ (taking a variable x to the tree $e^\dagger(x)$ and a parameter y to the trivial tree $\eta_Y(y)$). This extends to the unique homomorphism

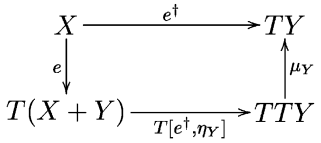
$$\hat{s} : T(X+Y) \to TY$$

of H-algebras taking a tree over $X+Y$ and substituting the leaves according to s. The property defining a solution, $e^\dagger$, is thus that the following triangle

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & TY \\ {\scriptstyle e}\downarrow & \nearrow_{\widehat{[e^\dagger, \eta_Y]}} & \\ T(X+Y) & & \end{array} \tag{1.2}$$

commutes. As mentioned above, T is a part of a monad, so that the substitution corresponding to $s : Z \to TY$ is given by $TZ \xrightarrow{Ts} TTY \xrightarrow{\mu_Y} TY$, where $\mu : TT \to T$ is the

monad multiplication. Thus, (1.2) is the following square

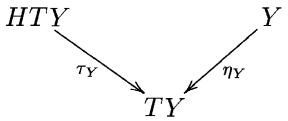


(1.3)

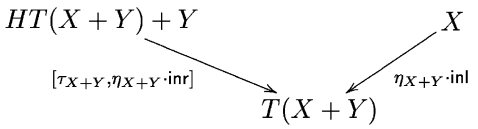
We are going to prove that "almost" all equations expressed by $e : X \to T(X + Y)$ have a unique solution $e^\dagger : X \to TY$. Exceptions are equations such as

$$x \approx x$$

What we want to avoid is that the right-hand side of an equation is a variable from X. This can be expressed categorically as follows: the final coalgebra TY is a fixed point of $H(\_) + Y$ (by Lambek's lemma [20]), therefore, TY is a coproduct of HTY and Y. Let us denote the coproduct injections by



where the right-hand injection is the unit of the monad T, and the left-hand one is the structure of an H-algebra mentioned in (b) above. The object $T(X + Y)$ is, thus, a coproduct of $HT(X + Y) + Y$ and X:

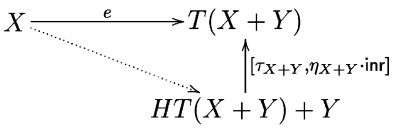


We can think of $HT(X + Y) + Y$ as the "rest" of $T(X + Y)$ when single variables from X have been removed. The equations we would like to solve are then the guarded ones:

Definition. By a *guarded equation morphism* is meant a morphism

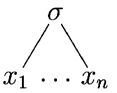
$$e : X \to T(X + Y)$$

(for an arbitrary object X "of variables" and an arbitrary object Y "of parameters") which factors through $HT(X + Y) + Y$:



Although guarded equation morphisms are allowed to have, on the right-hand sides, trees of arbitrary depth over X and Y, it is actually sufficient to solve *flat equations* where the right-hand sides are allowed to be only

(a) flat trees



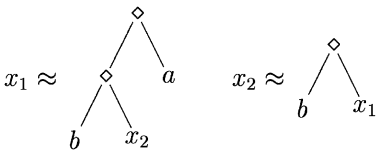
for an n-ary operation symbol σ and n variables $x_1,\ldots,x_n \in X$ (including $n=0$ where we have just σ)

or

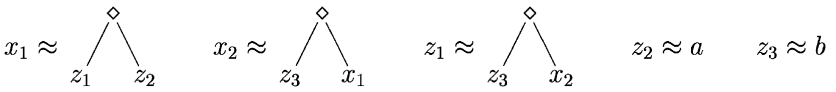
(b) single parameters from Y.

In fact, every guarded system can be "flattened" by adding auxilliary variables.

Example. To solve the following system



where $\diamond$ is a binary operation we flatten it by introducing new variables z_1, z_2, z_3 as follows:



Now for general functors H, *flat equation morphisms* have the form

$$e : X \to HX + Y.$$

But these are simply coalgebras of $H(\_) + Y$! And indeed, to solve e means precisely to use corecursion: a morphism $X \to TY$ is a solution of e iff it is the unique homomorphism from the coalgebra e into TY (the final coalgebra). This is our Solution Lemma, see Lemma 3.4.

The above flattening can also be performed quite generally, thus, the Solution Lemma implies the following

Solution Theorem. *Given an iteratable endofunctor, every guarded equation morphism has a unique solution.*

Now in [16] a theory (or monad) $\mathbb{T}$ on Set is called *completely iterative* provided that every guarded system of equations, $e : X \to T(X+Y)$, has a unique solution $e^\dagger : X \to TY$. Thus, our monad T is completely iterative. For example, if we start with a polynomial functor $H : \mathsf{Set} \to \mathsf{Set}$, then T is the monad of infinite properly labelled

trees. This is a free completely iterative monad on H, as proved in [16]. The proof there is very involved. We present here a considerably shorter and conceptually clearer proof. And moreover, the same proof works for all iteratable endofunctors of Set (not just the polynomial ones), in fact, all iteratable endofunctors of any category $\mathscr{A}$ with finite coproducts.

We can also view the completely iterative monad $T : \mathscr{A} \to \mathscr{A}$ as an object of the endofunctor category $[\mathscr{A}, \mathscr{A}]$. We prove that T is a final coalgebra of the following endofunctor $\hat{H}$ of $[\mathscr{A}, \mathscr{A}]$:

$$\hat{H}(B) = H \cdot B + Id \quad \text{for all } B : \mathscr{A} \to \mathscr{A}.$$

Now $[\mathscr{A}, \mathscr{A}]$ is a monoidal category whose tensor product $\otimes$ is composition and unit I is the identity functor Id. And the completely iterative monad generated by H is a monoid in $[\mathscr{A}, \mathscr{A}]$. We thus turn to the more general problem: given a monoidal category $\mathscr{B}$, we call an object H *iteratable* provided that the endofunctor $\hat{H} : \mathscr{B} \to \mathscr{B}$ given by $\hat{H}(B) = H \otimes B + I$ has a final coalgebra T. Assuming that binary coproducts of $\mathscr{B}$ distribute on the left with the tensor product, we deduce that T has a structure of a monoid, called the *completely iterative monoid generated by the object* H.

Throughout the paper we use the concept of category as "category in some universe". Thus, we can form, e.g., the category $[\mathscr{A}, \mathscr{A}]$ of all endofunctors for any category $\mathscr{A}$. As usual, a universe of "small sets" is supposed to be chosen, and the corresponding category is called Set. On two occasions we mention non-well-founded set theory briefly; there we denote by Class the category of classes and class functions.

Related work. The present paper is an expanded and improved version of the extended abstract [2].

In the very inspiring papers [24] and [25] of Moss, which we have discovered after completing [2], the Solution Theorem and Substitution Theorem we prove below have already been formulated and proved. In the setting of those papers, one works with final coalgebras of $H(\_ + X)$, but Moss already discussed in [24] the fact that these two approaches are equivalent; we state that explicitly below for the sake of completeness. Thus, the fact that the monad $\mathbb{T}$ we construct is completely iterative is due to Moss, whereas the result that $\mathbb{T}$ is free on H is new. And our proof of the complete iterativeness, presented here, is a happy combination of the proofs presented in [24] and [2].

The question of infinite trees forming a monad has been asked by Ghani and de Marchi, see also [17]. We acknowledge interesting discussion on that topic with them.

2. Iteratable functors

Assumption 2.1. Throughout this section, H denotes an endofunctor of a category $\mathscr{A}$ with finite coproducts. Whenever possible we denote by

$$\mathsf{inl} : X \to X + Y \quad \text{and} \quad \mathsf{inr} : Y \to X + Y$$

the first and the second coproduct injection respectively. Recall that, since coproducts are determined up to isomorphism only, equations such as $Z = X + Y$ are always meant as an isomorphism.

Remark 2.2. For the functor

$$H(\_) + X : \mathscr{A} \to \mathscr{A}$$

(i.e., for the coproduct of H with the constant functor of value X) it is well-known that

initial $(H(\_) + X)$-algebra $\equiv$ free H-algebra on X.

See e.g. [9]. More precisely, suppose that FX together with

$$\alpha_X : HFX + X \to FX$$

is an initial algebra of $H(\_) + X$. The components of α_X then form

an H-algebra $\varphi_X : HFX \to FX$

and

a universal arrow $\eta_X^F : X \to FX$.

That is, for every H-algebra

$$HA \to A$$

and for every morphism $f : X \to A$ there exists a unique homomorphism $f^\sharp : FX \to A$ of H-algebras with

$$f = f^\sharp \cdot \eta_X^F.$$

Example 2.3. *Polynomial endofunctors of* Set.

These are the endofunctors of the form

$$H_\Sigma Z = A_0 + A_1 \times Z + A_2 \times Z \times Z + \cdots = \coprod_{n<\omega} A_n \times Z^n,$$

where

$$\Sigma = (A_0, A_1, A_2, \ldots)$$

is a sequence of pairwise disjoint sets called the *signature*. An initial H-algebra can be described as the algebra of all finite Σ-labelled trees. Here a *Σ-labelled tree* t is represented by a partial function

$$t : \omega^* \to \bigcup_{n<\omega} A_n$$

whose domain of definition D_t is a nonempty and prefix-closed subset of ω^* (the set of all finite sequences of natural numbers), such that for any $i_1 i_2 \dots i_r \in D_t$ with $t(i_1 \dots i_r) \in A_n$ we have

$$i_1 i_2 \dots i_r i \in D_t \quad \text{iff} \quad i < n \quad (\text{for all } i < \omega).$$

The tree t is called finite if D_t is a finite set.

Now the functor

$$H_\Sigma(\_) + X$$

is also polynomial of signature

$$\Sigma_X = (X + A_0, A_1, A_2, \dots).$$

Therefore,

$$FX$$

can be described as the algebra of all finite Σ_X-labelled trees, i.e., trees with leaves labelled by variables or nullary operation symbols, and nodes with $n > 0$ successors labelled by n-ary operation symbols.

Remark 2.4.

(1) Dualizing the concept of a free H-algebra, we can study cofree H-coalgebras. A cofree H-coalgebra on an object X of $\mathscr{A}$ is just a free H^{op}-algebra on X in $\mathscr{A}^{op}$, where $H^{op} : \mathscr{A}^{op} \to \mathscr{A}^{op}$ is the obvious endofunctor. If $\mathscr{A}$ has finite products, then, by dualizing 2.2, we see that

final $(H(\_) \times X)$-coalgebra $\equiv$ cofree H-coalgebra on X.

Example: let H_Σ be a polynomial functor on Set. Then

$$H_\Sigma(\_) \times X$$

is also a polynomial functor, since

$$H_\Sigma Z \times X = \coprod_{n<\omega} X \times A_n \times Z^n.$$

This is the polynomial functor of signature

$$\Sigma^X = (X \times A_0, X \times A_1, X \times A_2, \dots).$$

A cofree H_Σ-coalgebra can be described as the coalgebra $\tilde{T}X$ of all (finite and infinite) Σ^X-labelled trees. Every node with n successors is labelled by (i) an n-ary operation symbol and (ii) a variable from X.

(2) Besides a free H-algebra on X and a cofree H-coalgebra on X, we have a third structure associated with X: a final coalgebra of $H(\_) + X$. We will show that it has an important universal property.

Definition 2.5. An endofunctor H of $\mathscr{A}$ is called *iteratable* provided that for every object X of $\mathscr{A}$ the endofunctor

$$H(\_) + X$$

has a final coalgebra.

Notation 2.6. Let

$$TX$$

denote a final coalgebra of $H(\_) + X$. The coalgebra map

$$\alpha_X : TX \to H(TX) + X$$

is, by Lambek's lemma [20], an isomorphism. Thus, TX is a coproduct of HTX and X; we denote the coproduct injections by

$$\tau_X : H(TX) \to TX \quad \text{and} \quad \eta_X : X \to TX.$$

Thus $[\tau_X, \eta_X] = \alpha_X^{-1} : H(TX) + X \to TX$.

In particular, TX is an H-algebra via τ_X.

Example 2.7. *Polynomial endofunctors of* Set *are iteratable.*

A final coalgebra

$$TX$$

of the (polynomial!) functor $H_\Sigma(\_) + X$ of signature Σ_X is the algebra of all finite and infinite Σ_X-labelled trees. That is, unlike the coalgebra

$$\tilde{T}X$$

of all Σ^X-labelled trees, see Remark 2.4, where every node carries a label from X and one from A_n (for the case of n children), the trees in TX have leaves labelled by variables or nullary operation symbols, and nodes with $n > 0$ successors labelled by n-ary operation symbols.

As a concrete example, consider a unary signature:

$$HZ = A \times Z.$$

We have defined three algebras for a set X of variables: the free algebra

$$FX = A^* \times X$$

of all finite Σ-labelled trees for $\Sigma = (\emptyset, A, \emptyset, \emptyset, \ldots)$, the cofree coalgebra

$$\tilde{T}X = (A \times X)^\infty$$

(where $(\_)^\infty$ denotes the set of all finite and infinite words in the given alphabet), and the coalgebra

$$TX = A^* \times X + A^\omega$$

(where $(\_)^\omega$ denotes the set of all infinite words in the given alphabet).

Example 2.8. *Generalized polynomial functors are iteratable.*

We want to include functors such as $HZ = Z^B$, where B is a (not necessarily finite) set; the description of TX is quite analogous to the preceding case. Here we introduce a *generalized signature* as a collection

$$\Sigma = (A_i)_{i \in Card}$$

of pairwise disjoint sets indexed by all cardinals such that for some cardinal λ we have

$$i \geqslant \lambda \quad \text{implies} \quad A_i = \emptyset.$$

(We say that Σ is a λ-*ary* generalized signature; the case $\lambda = \omega$ being the above one.) The generalized polynomial functor of generalized signature Σ is defined on objects by

$$H_\Sigma Z = \coprod_{j<\lambda} A_j \times Z^j$$

and analogously on morphisms.

An initial algebra of $H_\Sigma(\_) + X$, i.e., a free Σ-algebra, FX, on a set X of variables, can be described as the algebra of all *well-founded* Σ_X-labelled trees (i.e., Σ_X-labelled trees in which every branch is finite). For a λ-ary signature, a Σ_X-labelled tree can be formalized as follows: Let λ^* be the set of all words (= finite sequences) of ordinals smaller than λ. A Σ_X-labelled tree is a partial function

$$t : \lambda^* \to X + \coprod_{j<\lambda} A_j$$

defined on a nonempty, prefixed-closed subset D_t of λ^* such that for all $i_1 \dots i_r \in D_t$ we have: if $t(i_1 \dots i_r) \in X$, then $i_1 \dots i_r i \notin D_t$ for any i, and if $t(i_1 \dots i_r) \in A_j$, then

$$i_1 \dots i_r i \in D_t \quad \text{iff} \quad i < j \quad (\text{for all } i < \lambda).$$

The tree t is well-founded if D_t does not contain any infinite sequence of the form $i_1, i_1 i_2, i_1 i_2 i_3, \dots$, see, e.g., [9, II.3.6].

A *final coalgebra*, TX, of $H(\_)+X$ is, analogously to the finitary case, the coalgebra of *all* Σ_X-labelled trees, as proved, e.g., in [5].

Example 2.9. *Accessible* (= *bounded*) *endofunctors are iteratable.*

Recall that an endofunctor of Set is called accessible if it preserves λ-filtered colimits for some infinite cardinal λ. These are precisely the so-called bounded endofunctors, see [6]. This generalizes Examples 2.7 and 2.8 above.

Every accessible endofunctor has a final coalgebra: see a simple, explicit proof in [11, Proposition 1.3]. That proof applies, in fact, to accessible endofunctors of all locally presentable categories.

Since for H accessible also the functors $H(\_)+X$ are accessible, we conclude that

accessible $\Rightarrow$ iteratable.

Example 2.10. *Power-set functor and subfunctors.*

The power-set functor $\mathscr{P}:\mathsf{Set}\to\mathsf{Set}$ is not iteratable, in fact, it does not have a final coalgebra $T\emptyset$ (because there are no sets X isomorphic to $\mathscr{P}X$).

For every cardinal number κ the subfunctor $\mathscr{P}_\kappa$ of $\mathscr{P}$ defined on objects by

$$\mathscr{P}_\kappa Z=\{A\mid A\subseteq Z \text{ and } \operatorname{card} A<\kappa\}$$

is iteratable because it is accessible: for every cardinal λ with cofinality bigger than κ it is clear that $\mathscr{P}_\kappa$ preserves λ-filtered colimits.

For $\kappa=\aleph_0$ we use the notation $\mathscr{P}_f$. A final coalgebra of $\mathscr{P}_f$ has been described by Barr [11] as the coalgebra of all finitely-branching extensional trees (i.e., non-ordered trees such that any two distinct siblings yield non-isomorphic subtrees) modulo the following equivalence $\equiv$:

$t\equiv s$ iff for every $n\in\omega$ the cuttings $t|_n$ and $s|_n$ at level n have isomorphic extensional quotients.

This can be generalized to the following description of TX for $\mathscr{P}_f:TX$ is the coalgebra of all finitely-branching extensional trees with leaves labelled in $X+\{\emptyset\}$ modulo the above congruence $\equiv$ (where the cutting $t|_n$ is understood to have all new leaves labelled by $\emptyset$).

Example 2.11. *A non-accessible iteratable functor* $H:\mathsf{Set}\to\mathsf{Set}$ (*see Example 4.2 in* [6]).

We assume the Generalized Continuum Hypothesis (GCH) here. Let M be a class of cardinal numbers containing 1. Define

$$\mathscr{P}_M:\mathsf{Set}\to\mathsf{Set}$$

on sets A by

$$\mathscr{P}_M A=\{B\subseteq A\mid B=\emptyset \text{ or } \operatorname{card}(B)\in M\}$$

and on functions $f:A\to A'$ by

$$\mathscr{P}_M f:B\mapsto\begin{cases} f[B] & \text{if } f \text{ restricted to } B \text{ is one-to-one,}\\ \emptyset & \text{otherwise.}\end{cases}$$

Then $\mathscr{P}_M$ is accessible iff M is a set. In fact, if M is a set with supremum smaller than λ, then $\mathscr{P}_M$ preserves λ-filtered colimits; if M is a proper class then $\mathscr{P}_M$ does not preserve λ-filtered colimits for any λ.

Now let M be a proper class of cardinals such that there exist arbitrarily large regular cardinals α with the property

$$\alpha \notin M \quad \text{and} \quad 2^{\alpha} \notin M. \tag{2.1}$$

Then the following lemma shows that the functor $\mathscr{P}_M(\_) + X$ has, for every set X, "fixed points" α and 2^{α}, where $\alpha \geqslant \mathrm{card}(X)$ is any regular cardinal number with $\alpha \notin M$ and $2^{\alpha} \notin M$. It follows from [5] that, then, a final coalgebra of $\mathscr{P}_M(\_) + X$ exists, i.e., that $\mathscr{P}_M$ is iteratable (but not accessible). For the proof of the lemma we use the following result: if α is a regular, infinite cardinal number and $\beta < \alpha$, then $\alpha^{\beta} = \alpha$ (under GCH), see [19].

Lemma. *Let X be a set and $\alpha \notin M$ an infinite regular cardinal number with* $\mathrm{card}(X) \leqslant \alpha$. *Then every set A of cardinality α is a "fixed point" of $\mathscr{P}_M(\_) + X$, i.e.,*

$$A \cong \mathscr{P}_M(A) + X.$$

Proof. Since $1 \in M$, we have $\mathrm{card}(\mathscr{P}_M(A)) \geqslant \mathrm{card}(A)$, thus, it is sufficient to prove

$$\mathrm{card}(A) \geqslant \mathrm{card}(\mathscr{P}_M(A) + X).$$

Since $\mathrm{card}(A) = \alpha \notin M$, we have

$$\mathscr{P}_M(A) \subseteq \bigcup_{\beta < \alpha} \{B \subseteq A \mid \mathrm{card}(B) = \beta\}$$

therefore

$$\mathrm{card}(\mathscr{P}_M(A) + X) \leqslant \left(\sum_{\beta < \alpha} \alpha^{\beta}\right) + \mathrm{card}(X)$$

$$\leqslant \left(\sum_{\beta < \alpha} \alpha\right) + \alpha = \alpha \times \alpha + \alpha = \alpha. \qquad \square$$

Example 2.12. Iteratable endofunctors of Set do not have desired stability properties. For example, if F and G are iteratable, then neither $F \cdot G$ nor $F + G$ need to be iteratable. In fact, in the notation of Example 2.11, consider classes M and M' of cardinal numbers containing 1 and such that

(1) $M \cup M'$ is the class of all cardinal numbers

(2) there exist arbitrarily large cardinals α with $\alpha \notin M$ and $2^{\alpha} \notin M$

and

(3) there exist arbitrarily large cardinals β with $\beta \notin M'$ and $2^{\beta} \notin M'$

Then $\mathscr{P}_M$ and $\mathscr{P}_{M'}$ are both iteratable by Example 2.11. But $\mathscr{P}_M + \mathscr{P}_{M'}$ does not have any fixed point (for every set A either $\mathrm{card}(\mathscr{P}_M A) > \mathrm{card}(A)$, or $\mathrm{card}(\mathscr{P}_{M'} A) > \mathrm{card}(A)$), hence, $\mathscr{P}_M + \mathscr{P}_{M'}$ it is not iteratable, having no final coalgebra. Analogously with $\mathscr{P}_M \cdot \mathscr{P}_{M'}$.

Example 2.13. All set functors are "almost" iteratable. There are, of course, non-iteratable endofunctors of Set, e.g., the power-set functor $\mathscr{P}$. However, every functor $H : \mathsf{Set} \to \mathsf{Set}$ can be extended (uniquely up to natural isomorphism) to an endofunctor H^∞ of Class, the category of all large sets (= classes) and functions so that H^∞ preserves colimits, of transfinite chain, see [11].

Applying this to $H(\_) + X$ we see that a final coalgebra, TX, always exists, but it can be a proper class.

Example 2.14. *Power-set functor in non-well-founded set theory.*

The power-set functor $\mathscr{P} : \mathsf{Class} \to \mathsf{Class}$ (assigning to every class the class of its sub*sets*) is iteratable. Assuming the anti-foundation axiom (AFA), for every class X we can describe TX as the so called hyperuniverse of sets built up using the elements of X as atoms. In Chapter 1 of [1] the sets of this hyperuniverse were called the X-sets and they form the class V_X of [12]. The Substitution and Solution theorems have been exploited in the context of these hyperuniverses by applying them to Milner's CCS approach to concurrency, the Liar Paradox and Situation Theory. See also [13].

Example 2.15. *Continuous functors are iteratable.*

Recall that a functor is called *continuous* if it preserves limits of ω^{op}-sequences.

Here we assume that our base category $\mathscr{A}$ has

1. a terminal object 1
2. limits of ω^{op}-sequences

and

3. binary coproducts commuting with ω^{op}-limits.

(Set fulfills these requirements, of course.) Every continuous endofunctor F has a final coalgebra $\lim_{n<\omega} F^n 1$—this is dual to the famous construction of an initial algebra as $\mathrm{colim}_{n<\omega} F^n 0$ first formulated in [3].

If H is continuous, then due to 3., all functors $H(\_) + X$ are continuous, thus, have a final coalgebra

$$TX = \lim_{n<\omega} (H(\_) + X)^n 1.$$

Remark 2.16. Denote by $U : H\text{-}\mathsf{Alg} \to \mathscr{A}$ the forgetful functor of the category of all H-algebras and homomorphisms. The universal property of free H-algebras $\varphi_X : HFX \to FX$ (provided they exist on all objects X of $\mathscr{A}$) makes U a right adjoint. The left adjoint is the functor

$$X \mapsto (FX, \varphi_X).$$

We now show a related universal property of the H-algebras $\tau_X : HTX \to TX$ of 2.6: given a morphism $s : X \to TY$ we prove that there is a unique homomorphism $\hat{s} : TX \to TY$ of H-algebras extending s. This is interesting even for the basic case of the polyno-

mial endofunctors of Set: here a morphism $s : X \to TY$ can be viewed as a substitution rule, substituting a variable $x \in X$ by the Σ_Y-labelled tree $s(x)$. We obviously have a homomorphism $\hat{s} : TX \to TY$ extending s: take a tree $t \in TX$, substitute every variable $x \in X$ on any leaf of t by the tree $s(x)$ and obtain a tree

$$t' = Ts(t) \quad \text{in } TTY$$

over TY. Now forget that t' is a tree of trees and obtain a tree $\hat{s}(t)$ in TY. However, it is not obvious that such a homomorphism is unique. This is what we prove now:

Substitution Theorem 2.17. *For every iteratable endofunctor H of $\mathscr{A}$ and any morphism*

$$s : X \to TY \quad \text{in } \mathscr{A}$$

there exists a unique extension into a homomorphism

$$\hat{s} : TX \to TY$$

of H-algebras. That is, a unique homomorphism $\hat{s} : (TX, \tau_X) \to (TY, \tau_Y)$ with $s = \hat{s} \cdot \eta_X$.

Proof. We turn $TX + TY$ into a coalgebra of type $H(\_) + Y$ as follows: the coalgebra map is

$$TX+TY = HTX+X+TY \xrightarrow{id+[s,id]} HTX+TY = HTX+HTY+Y \xrightarrow{[H\mathsf{inl},H\mathsf{inr}]+id} H(TX+TY)+Y$$

There exists a unique homomorphism

$$f : TX + TY \to TY$$

of $(H(\_) + Y)$-coalgebras. Equivalently, a unique morphism

$$f = [f_1, f_2] : TX + TY \to TY$$

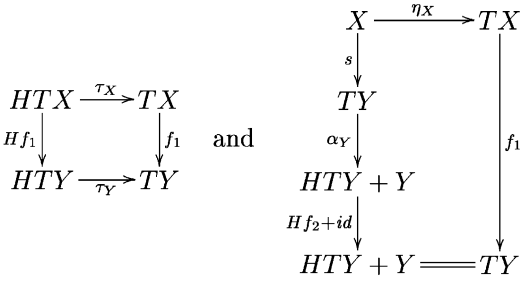
in $\mathscr{A}$ for which the following two squares

$$\begin{array}{ccc} HTX + X & = & TX \\ {\scriptstyle [\mathsf{inl}\cdot H\mathsf{inl},(H\mathsf{inr}+id)\cdot s]}\downarrow & & \\ H(TX + TY) + Y & & \downarrow{\scriptstyle f_1} \\ {\scriptstyle Hf+id}\downarrow & & \\ HTY + Y & = & TY \end{array} \qquad \begin{array}{ccc} HTY + Y & = & TY \\ \downarrow{\scriptstyle Hf_2+id} & & \downarrow{\scriptstyle f_2} \\ HTY + Y & = & TY \end{array}$$

commute. The right-hand square shows that f_2 is an endomorphism of the final $(H(\_)+Y)$-coalgebra—thus,

$$f_2 = id.$$

The left-hand square is equivalent to the commutativity of the following two squares:



The square on the left tells us that f_1 is a homomorphism of H-algebras. And since $f_2 = id$ (thus $Hf_2 + id = id$) and $\alpha_Y^{-1} = [\tau_Y, \eta_Y]$, the square on the right states $f_1 \cdot \eta_X = s$, i.e., f_1 extends s. This proves that there is a unique extension of s to a homomorphism: put $\hat{s} = f_1$. □

Corollary 2.18. *The formation of TX and η_X (for all objects X) and of $\hat{s}$ (for all morphisms $s : X \to TY$) is a Kleisli triple on $\mathscr{A}$.*

In fact, the axioms of Kleisli triples (i.e., $\hat{s} \cdot \eta_X = s$, $\widehat{\eta_X} = id$, and $\hat{s} \cdot \hat{t} = \widehat{\hat{s} \cdot t}$) follow immediately from the uniqueness of $\hat{s}$ in the Substitution Theorem.

In other words, TX is the object part of a functor T, such that η_X are the components of a natural transformation $\eta : Id \to T$, and we have a natural transformation $\mu : TT \to T$ defined by

$$\mu_X = \widehat{id} : TTX \to TX$$

forming a monad $\mathbb{T} = (T, \eta, \mu)$ on $\mathscr{A}$. Observe that

μ_X is a homomorphism of H-algebras

since each $\hat{s}$ is. Also, for every morphism $f : A \to B$ in $\mathscr{A}$, $Tf : TA \to TB$ is a homomorphism of H-algebras (because $Tf = \widehat{\eta_B \cdot f}$). Thus,

$$\tau : HT \to T$$

is a natural transformation.

Remark 2.19. Our Substitution Theorem has been proved by Moss in [24] as Lemma 2.4, except that he works with final coalgebras of $H(\_ + X)$ rather than of $H(\_) + X$. However, in a remark preceding his 2.4 he shows the following:

Lemma. *An endofunctor H is iteratable iff for every object X the endofunctor $H(\_ + X)$ has a final coalgebra. In fact*

(i) *a final coalgebra of $H(\_ + X)$ is HTX with the structure map*

$$H\alpha_X : HTX \to H(HTX + X)$$

and, conversely,

(ii) *if* $\hat{\alpha}_X : \hat{T}X \to H(\hat{T}X + X)$ *is a final* $H(\_ + X)$*-coalgebra, then* $\hat{T}X + X$ *with the structure map*

$$\hat{\alpha}_X + X : \hat{T}X + X \to H(\hat{T}X + X) + X$$

is a final coalgebra for $H(\_) + X$.

Proof. Ad (i): given an $H(\_ + X)$-coalgebra

$$\rho : R \to H(R + X)$$

consider the $(H(\_) + X)$-coalgebra

$$\rho + id : R + X \to H(R + X) + X$$

The unique $(H(\_) + X)$-homomorphism $h : R + X \to TX = HTX + X$ has the form $h = h_1 + id_X$ where $h_1 : R \to HTX$ yields the desired $H(\_ + X)$-homomorphism.

Ad (ii): given an $(H(\_) + X)$-coalgebra

$$\rho : R \to HR + X$$

consider the $H(\_ + X)$-coalgebra

$$H\rho : HR \to H(HR + X).$$

The unique $H(\_ + X)$-homomorphism $h : HR \to \hat{T}X$ yields the desired unique $(H(\_) + X)$-homomorphism $g : R \to \hat{T}X + X$ as follows

$$g \equiv R \xrightarrow{\rho} HR + X \xrightarrow{h+id} \hat{T}X + X. \qquad \square$$

Remark 2.20. Note that the last result is an instance of a general fact about categories of fixed points of functors. Indeed, suppose that $F, G : \mathscr{A} \to \mathscr{A}$ are endofunctors. Then applying F and G respectively yields functors

$$FG\text{-}\mathsf{Coalg} \underset{F}{\overset{G}{\rightleftarrows}} GF\text{-}\mathsf{Coalg}$$

which preserve fixed points (i.e., coalgebras whose structure maps are isomorphisms). It is trivial to show that the restrictions of the latter to the full subcategories of fixed points of F-Coalg and G-Coalg respectively are equivalences of categories that are inverse to one another.

Definition 2.21. The above monad $\mathbb{T}$, associated with any iteratable endofunctor H, is called the *completely iterative monad generated by* H.

Examples 2.22.

(1) The completely iterative monad generated by the endofunctor

$$HZ = A \times Z$$

of Set is the monad

$$TX = A^* \times X + A^\omega.$$

This can be described as the free-algebra monad of the variety of algebras with
(a) unary operations f_a for $a \in A$,
(b) nullary operations indexed by A^ω (i.e., constants of the names $a_0a_1a_2\ldots \in A^\omega$), and
(c) satisfying the equations

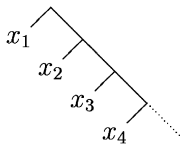
$$f_a(a_0a_1a_2\ldots) = aa_0a_1a_2\ldots \quad \text{for all } a, a_0, a_1, \ldots \in A$$

In this case, $\mathbb{T}$ is a finitary monad on Set.

(2) The completely iterative monad generated by the endofunctor

$$HZ = Z \times Z$$

of Set is the monad TX of all binary trees with leaves indexed in X. This is not finitary: consider the following element of TX:



in which all x_i are pairwise distinct.

(3) Let

CPO

denote the category of CPO's (say, posets with a smallest element $\perp$ and joins of ω-chains) and strict continuous functions (i.e., those preserving $\perp$ and joins of ω-chains). For all *locally continuous* functors $H : \mathsf{CPO} \to \mathsf{CPO}$, i.e., such that the derived functions

$$\mathsf{CPO}(A,B) \to \mathsf{CPO}(HA,HB), \quad f \mapsto Hf$$

are all continuous, it is well-known that
initial H-algebra $\equiv$ final H-coalgebra,
see [26]. Since each $H(\_) + X$ is also locally continuous, we deduce that
locally continuous functors are iteratable,
and in this case

$$FX \equiv TX$$

that is, the completely iterative monad $\mathbb{T}$ is just the free algebra monad $\mathbb{F}$ on H.

(4) Analogously for the category

CMS

of all complete metric spaces and contractions: every *contractive* endofunctor $H : \mathsf{CMS} \to \mathsf{CMS}$, i.e., such that the derived functions

$$\mathsf{CMS}(A,B) \to \mathsf{CMS}(HA,HB), \quad f \mapsto Hf$$

are all contractive with a common constant <1, has a single fixed point. Therefore,

initial H-algebra $\equiv$ final H-coalgebra,

see [7]. Since each $H(\_) + X$ is also locally contractive, we again get

$$\mathbb{T} = \mathbb{F}.$$

Remark 2.23.

(1) The Kleisli category

$$\mathscr{A}_{\mathbb{T}} \to \mathscr{A}$$

of the completely iterative monad is the above category $\mathscr{K}$ of all H-algebras $\tau_X : HTX \to TX$ (with its forgetful functor $\mathscr{K} \to \mathscr{A}$). This follows from the Substitution Theorem.

(2) The Eilenberg–Moore category

$$\mathscr{A}^{\mathbb{T}} \to \mathscr{A}$$

of all $\mathbb{T}$-algebras and $\mathbb{T}$-homomorphisms seems to be a new construct. As seen in 2.22, it is usually infinitary.

3. Solution theorem

3.1. Recall from the Introduction that a *solution* of an equation morphism $e : X \to T(X+Y)$ is a morphism $e^\dagger : X \to TY$ such that the following square

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & TY \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu_Y} \\ T(X+Y) & \xrightarrow[T[e^\dagger,\eta_Y]]{} & TTY \end{array}$$

commutes. Elgot used the language of algebraic theories, i.e., Kleisli categories, rather than monads. Both equations and solutions are morphisms of the Kleisli category, here:

$$e : X \to X + Y \quad \text{and} \quad e^\dagger : X \to Y.$$

If we denote by $*$ the composition of the Kleisli category (i.e., $g * f = \mu_Z \cdot Tg \cdot f$ for $f : X \to TY$ and $g : Y \to TZ$ in $\mathscr{A}$) then a solution $e^\dagger$ is defined by the equality

$$e^\dagger = [e^\dagger, 1] * e.$$

This is the definition used in [15,16]. We are not going to use this notation below.

Recall further from the Introduction that a *flat equation morphism*

$$e : X \to HX + Y$$

is just another name for a coalgebra of $H(\_) + Y$. However, we can also view e as a guarded equation morphism. More precisely, we denote by

$$\rho_{X,Y} : HX + Y \to T(X + Y)$$

the "natural connecting morphism" whose left-hand component is

$$HX \xrightarrow{H\eta X} HTX \xrightarrow{HT\mathsf{inl}} HT(X + Y) \xrightarrow{\tau_{X+Y}} T(X + Y)$$

and the right-hand one is

$$Y \xrightarrow{\mathsf{inr}} X + Y \xrightarrow{\eta_{X+Y}} T(X + Y).$$

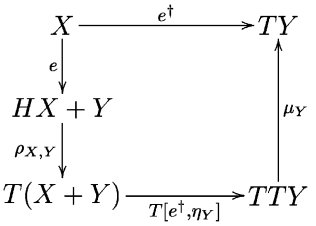
Since $\rho_{X,Y}$ factors through $[\tau_{X+Y}, \eta_{X+Y}\mathsf{inr}]$, we see that

$$\rho_{X+Y} e : X \to T(X + Y)$$

is a guarded equation morphism. We denote, for short, by

$$e^\dagger : X \to TY$$

a solution of $\rho_{X,Y} e$ (whenever there is no danger of confusion). Explicitly, $e^\dagger$ is a morphism such that the following diagram



commutes.

Examples 3.2.

(1) For polynomial functors solutions of flat equations are discussed in the Introduction.

(2) For the finite-power-set functor $\mathscr{P}_f : \mathsf{Set} \to \mathsf{Set}$ a flat system of equations without parameters has the following form

$$x_1 \approx A_1$$

$$x_2 \approx A_2$$

$$\vdots$$

for a set $X=\{x_1,x_2,\ldots\}$ of variables, where A_1, A_2, ... are finite subsets of X. This is the concept of a flat system of equations as used in non-well-founded set theory.

The functor $\mathscr{P}_f$ is iteratable, see Example 2.10. In non-well-founded set theory, a final coalgebra $T\emptyset$ is described as the coalgebra of all hereditarily finite sets, see [13]. Thus, every solution of equation systems as above is found in that coalgebra. In well-founded set theory, solutions will be extensional trees modulo the equivalence described in Example 2.10.

(3) The power-set functor $\mathscr{P}$ leads to flat systems of equations without parameters of the form above, except that here the subsets A_1, A_2, ... of X are arbitrary, not necessarily finite. The possibility of having a unique solution for every flat system of equations is (one of the formulations of) the anti-foundation axiom leading to non-well-founded set theory, see [1,13].

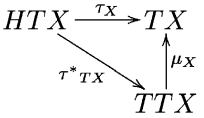
Notation 3.3. We denote by

$$\tau^* : H \to T$$

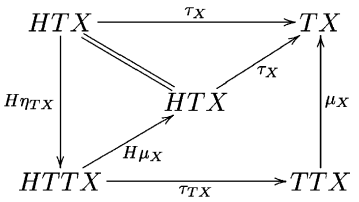
the composite

$$H \xrightarrow{H\eta} HT \xrightarrow{\tau} T.$$

Observe that the following triangle



commutes for every object. This follows from μ_X being a homomorphism of H-algebras and $\mu \cdot \eta T = id$:

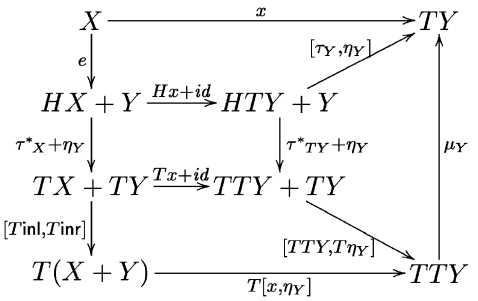


Solution Lemma 3.4. *For flat equation morphisms we have*

$$solution = corecursion.$$

That is, a flat equation morphism $e : X \to HX + Y$ has a unique solution, viz, the unique homomorphism of the coalgebra e into the final coalgebra TY of $H(\_) + Y$.

Proof. For any morphism $x : X \to TY$, consider the following diagram



The lower square and the middle one clearly commute. Also the right-hand square commutes by 3.3. Now suppose we put $e^\dagger$ in the place of x in the diagram. Then the outer square commutes, and therefore the upper square does, which shows that $e^\dagger$ is an $H(\_) + Y$ coalgebra homomorphism, and thus $e^\dagger = \tilde{e}$, where $\tilde{e}$ denotes the unique homomorphism into the final coalgebra TY.

Conversely, if $\tilde{e}$ is put in the place of x, then the upper square commutes and thus the whole diagram does, which shows that $\tilde{e}$ is a solution for e. □

Remark 3.5. In the Introduction we have mentioned that every guarded equation morphism $e : X \to T(X + Y)$ has a "flattening" by introducing additional variables, Z. That is, there is a flat equation morphism

$$g : X + Z \to H(X + Z) + Y$$

such that to solve e is "the same" as to solve g. This is, in fact, a general phenomenon:

Proposition 3.6. *For every guarded equation morphism*

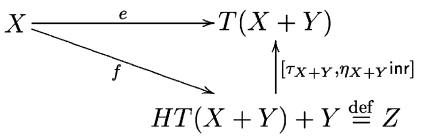
$$e : X \to T(X + Y)$$

there exists a flat equation morphism

$$g : X + Z \to H(X + Z) + Y$$

such that the left-hand component of $g^\dagger : X + Z \to TY$ *is a solution of* e.

Proof. Since e is guarded, we have a commutative triangle



The above object Z has the property that

$$X + Z = T(X + Y).$$

More precisely, $T(X+Y)$ is a coproduct of X and Z with injections

$$X \xrightarrow{\eta_{X+Y}\mathsf{inl}} T(X+Y)$$

and

$$Z = HT(X+Y)+Y \xrightarrow{id+\mathsf{inr}} HT(X+Y)+(X+Y) = T(X+Y)$$

respectively. The morphism g we are to define thus has the codomain $HT(X+Y)+Y=Z$. Put simply

$$g = [f, id] : X + Z \to Z.$$

The solution $g^\dagger : X + Z = X + HT(X+Y) + Y \to TY$ has components $h_1 : X \to TY$, $h_2 : HT(X+Y) \to TY$ and $h_3 : Y \to TY$. The property of being a solution means, by the Solution Lemma, precisely that $[h_1, h_2, h_3] : T(X+Y) \to TY$ is a homomorphism of coalgebras. That is, $g^\dagger$ is a solution if and only if the following square

$$\begin{array}{ccc} T(X+Y) = X + HT(X+Y) + Y & \xrightarrow{[f, HT(X+Y)+Y]} & HT(X+Y)+Y \\ \downarrow{\scriptstyle [h_1,h_2,h_3]} & & \downarrow{\scriptstyle H[h_1,h_2,h_3]=Hg^\dagger+id} \\ TY & \xrightarrow[[\tau_Y,\eta_Y]]{} & HTY + Y \end{array}$$

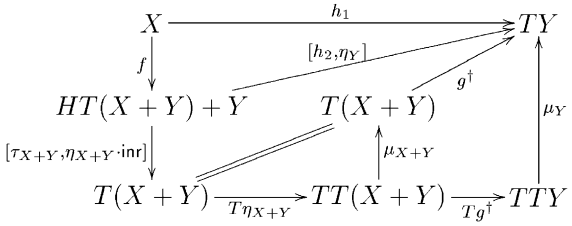
commutes. Equivalently, iff the following hold:

$$h_3 = \eta_Y$$

$$h_2 = \tau_Y \cdot Hg^\dagger$$

$$h_1 = [\tau_Y, \eta_Y] \cdot (Hg^\dagger + id) \cdot f = [h_2, \eta_Y] \cdot f.$$

We prove that h_1 solves e. Since $g^\dagger \cdot \eta_{X+Y} = [h_1, h_3] = [h_1, \eta_Y]$ and $e = [\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}] \cdot f$ we are to prove the commutativity of the outward square in the following diagram



The right-hand inner square commutes because $g^\dagger$ is a homomorphism of H-algebras: $g^\dagger \cdot \tau_{X+Y} = h_2 = \tau_Y \cdot Hg^\dagger$ and thus, by Substitution Theorem it is enough to observe that

$$(g^\dagger \cdot \mu_{X+Y}) \cdot \eta_{T(X+Y)} = g^\dagger = \mu_Y \cdot \eta_{TY} \cdot g^\dagger = (\mu_Y \cdot Tg^\dagger) \cdot \eta_{T(X+Y)}.$$

All the other inner parts also commute (e.g., $g^\dagger \cdot [\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}] = [h_2, h_3] = [h_2, \eta_Y]$). □

Remark 3.7. The proof of the preceding proposition gives more than the statement: every solution $e^\dagger$ of the original equation morphism yields a solution of the flat one by the rule

$$g^\dagger \equiv X + Z = T(X + Y) \xrightarrow{T[e^\dagger, \eta_Y]} TTY \xrightarrow{\mu_Y} TY.$$

In fact, the morphism

$$\mu_Y \cdot T[e^\dagger, \eta_Y] : X + HT(X + Y) + Y \to TY$$

has the following components

$$h_3 = \mu_Y \cdot T[e^\dagger, \eta_Y] \cdot \eta_{X+Y} \cdot \mathsf{inr} = \mu_Y \cdot \eta_{TY} \cdot \eta_Y = \eta_Y$$

(by naturality: $T[e^\dagger, \eta_Y] \cdot \eta_{X+Y} = \eta_{TY} \cdot [e^\dagger, \eta_Y]$)

$$h_2 = \mu_Y \cdot T[e^\dagger, \eta_Y] \cdot \tau_{X+Y} = \mu_Y \cdot \tau_{TY} \cdot HT[e^\dagger, \eta_Y] = \tau_Y \cdot H\mu_Y \cdot HT[e^\dagger, \mu_Y] = \tau_Y \cdot Hg^\dagger$$

(since $T(\_)$ and μ_Y are homomorphisms of H-algebras), and

$$h_1 = \mu_Y \cdot T[e^\dagger, \eta_Y] \cdot \eta_{X+Y} \cdot \mathsf{inl} = \mu_Y \cdot \eta_{TY} \cdot e^\dagger = e^\dagger : X \to TY.$$

Moreover, by definition of $(\_)^\dagger$ for $e = [\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}] \cdot f$,

$$h_1 = e^\dagger = \mu_Y \cdot T[e^\dagger, \eta_Y] \cdot [\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}] \cdot f = [h_2, \eta_Y] \cdot f.$$

Thus, the three equations of the above proof hold, i.e., $g^\dagger$ is a homomorphism of $(H(\_) + Y)$-coalgebras.

Corollary 3.8 (Solution Theorem). *Given an iteratable functor, every guarded equation morphism has a unique solution.*

Remark. This is the result called Parametric Corecursion by Moss, see [24] We have proved it, independently, in [2].

Proof. In fact, the existence follows from 3.4 and 3.6. The uniqueness from 3.7: since $g^\dagger = \mu_Y \cdot T[e^\dagger, \eta_Y]$ implies $g^\dagger \cdot \eta_{X+Y} = \mu_Y \cdot \eta_{TY} \cdot [e^\dagger, \eta_Y] = [e^\dagger, \eta_Y]$ we have $e^\dagger = g^\dagger \cdot \eta_{X+Y} \cdot \mathsf{inr}$. Thus, the uniqueness of $g^\dagger$ (see 3.4) proves the uniqueness of $e^\dagger$. □

4. Completely iterative monads

Assumption 4.1. In the present section we assume that a category $\mathscr{A}$ with finite coproducts is given such that coproduct injections are monomorphisms. (One can work, more generally, with binary coproducts without further restriction, see Remark 4.16 below.)

We are going to introduce solutions of guarded equations in general monads, and obtain the concept of complete iterativity for monads. Our main result will be that the above monad $\mathbb{T}$ is a free completely iterative monad on the given functor H.

Elgot has introduced the concept of an *ideal algebraic theory* in order to speak about ideal equations and (completely) iterative theories. As we show below, his concept is the special case, for $\mathscr{A} = \mathsf{Set}$ and for finitary monads, of the following:

Definition 4.2. A monad $\mathbb{S} = (S, \eta, \mu)$ on $\mathscr{A}$ is called *ideal* provided that
(i) S is a coproduct of endofunctors, $S = S' + Id$, with $\eta = \mathsf{inr} : Id \to S$
and
(ii) $\mu : SS \to S$ restricts to $\mu' : S'S \to S'$.

Remark 4.3. More precisely, we should say that an ideal monad is a sixtuple $(S, \eta, \mu, S', \sigma, \mu')$ consisting of a monad (S, η, μ), a natural transformation $\sigma : S' \to S$ forming inl of the coproduct $S = S' + Id$ with $\eta = \mathsf{inr}$, and a natural transformation $\mu' : S'S \to S'$ such that the following square (expressing "a restriction of μ")

$$\begin{array}{ccc} S'S & \xrightarrow{\sigma S} & SS \\ {\scriptstyle \mu'}\downarrow & & \downarrow{\scriptstyle \mu} \\ S' & \xrightarrow[\sigma]{} & S \end{array}$$

commutes.

However, the above definition is precise enough since we assume that coproduct injections in $\mathscr{A}$ (and, thus, in $[\mathscr{A}, \mathscr{A}]$) are monomorphisms, which makes μ' unique.

Examples 4.4.

(1) The completely iterative monad $\mathbb{T}$ for a given iteratable endofunctor H, see Definition 2.21, is ideal. Here

$$T = HT + Id$$

with coproduct injections τ and η. And for $\mu' = H\mu$ the relevant square commutes, because each $\mu_X : TTX \to TX$ is (by definition) a homomorphism of H-algebras:

$$\begin{array}{ccc} HTTX & \xrightarrow{\tau_{TX}} & TTX \\ {\scriptstyle H\mu_X}\downarrow & & \downarrow{\scriptstyle \mu_X} \\ HTX & \xrightarrow[\tau_X]{} & TX \end{array}$$

(2) Consider the variety of algebras on one binary operation given by the single equation

$$(xy)z = x.$$

The corresponding monad $\mathbb{S}$ is easily seen to be such that $\eta : Id \to S$ is a coproduct injection. However, this monad is not ideal: this follows from the fact that although none of the terms

$$t = \overset{u}{\wedge}\, z \quad \text{and} \quad s = x \overset{\wedge}{} y$$

is congruent to a variable, the term $t[s/u]$ is congruent to x.

Remark 4.5. The definition of ideal theory used by Elgot is the following. An *algebraic theory* (in the sense of Lawvere) is a category whose objects are given by the set $\mathbb{N}$ of natural numbers and such that for each $n \geq 0$ there are so-called *distinguished morphisms*

$$i_1, \ldots, i_n : 1 \to n$$

which form coproduct injections. Such a theory is called *ideal* whenever the following property holds: if $f : 1 \to n$ is not distinguished, then $g \cdot f : 1 \to m$ is not distinguished for every $g : n \to m$. Recall that every finitary variety gives rise to an algebraic theory as follows: an arrow

$$s : n \to m$$

is a substitution that gives for each of n variables $x_1, \ldots, x_n$ a term $s(x_i)$ in m variables. The distinguished morphism

$$i_k : 1 \to n$$

substitutes x_k for the given variable.

Recall further that finitary varieties correspond to finitary monads on Set. Moreover, for every finitary variety, the notion of ideal monad as defined in 4.2 coincides with the notion of ideal theory:

Lemma 4.6. *The algebraic theory corresponding to a finitary variety $\mathscr{V}$ is ideal if and only if the finitary monad corresponding to $\mathscr{V}$ is ideal.*

Proof. Suppose the theory of a given finitary variety is ideal. Let $(S, \eta, s \mapsto \hat{s})$ be the corresponding finitary monad given by its Kleisli triple. Then for arbitrary finite sets X, Y and substitution $s : X \to SY$, the homomorphism $\hat{s} : SX \to SY$ satisfies the following property: if $t \in SX$ is not (congruent to) a variable, then neither is $\hat{s}(t) \in SY$. In particular, this is true for $Sf = \widehat{f \cdot \eta_Y}$ for any $f : X \to Y$. Since Sf preserves variables, we conclude that $S = S' + Id$ with coproduct injection $\eta : Id \to S$ (for infinite sets use that S is finitary). That μ restricts to μ' follows since $\mu_Y = \widehat{id_{SY}}$.

Conversely, suppose that the finitary monad (S, η, μ) of a given variety $\mathscr{V}$ is ideal in the sense of Definition 4.2. Let $s : X \to SY$ be any substitution where X and Y are finite, and let $t \in S'X$. Then $\hat{s}(t)$ is in $S'Y$ since $\hat{s} = \mu_Y \cdot Ss$, which on $S'X$ restricts to $\mu'_Y \cdot S's$. But this is equivalent to the theory of $\mathscr{V}$ being ideal in the sense of Elgot. □

Definition 4.7. Let $\mathbb{S}$ be an ideal monad on $\mathscr{A}$.

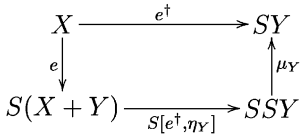
(1) By an *equation morphism* we understand a morphism in $\mathscr{A}$ of the form

$$e : X \to S(X + Y), \quad X, Y \text{ are objects of } \mathscr{A}.$$

(2) By a *solution* of e is understood a morphism

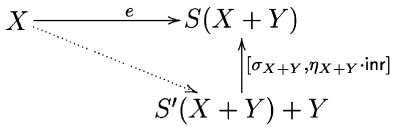
$$e^\dagger : X \to SY$$

for which the following diagram



commutes.

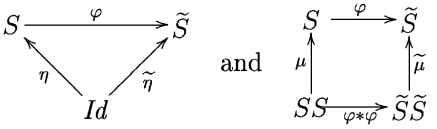
(3) We call *e guarded* if it factors through $S'(X+Y)+Y$:



Definition 4.8. An ideal monad is called *completely iterative* provided that every guarded equation morphism has a unique solution.

Example 4.9. The monad $\mathbb{T}$ associated with an iteratable functor H is completely iterative. This is the Solution Theorem.

We are going to prove that solutions are preserved by monad morphisms. Recall that for monads $\mathbb{S}=(S,\eta,\mu)$ and $\tilde{\mathbb{S}}=(\tilde{S},\tilde{\eta},\tilde{\mu})$ a *monad morphism* $\varphi:\mathbb{S}\to\tilde{\mathbb{S}}$ is a natural transformation $\varphi:S\to\tilde{S}$ such that the following diagrams



commute. (Here, $\varphi*\varphi$ denotes the horizontal composition, i.e., $\varphi*\varphi=\varphi\tilde{S}\cdot S\varphi=\tilde{S}\varphi\cdot\varphi S$.)

Definition 4.10. If $\mathbb{S}$ and $\tilde{\mathbb{S}}$ are ideal monads, we call a morphism $\varphi:\mathbb{S}\to\tilde{\mathbb{S}}$ *ideal* if it has the form $\varphi=\varphi'+id$ for a natural transformation $\varphi':S'\to\tilde{S}'$.

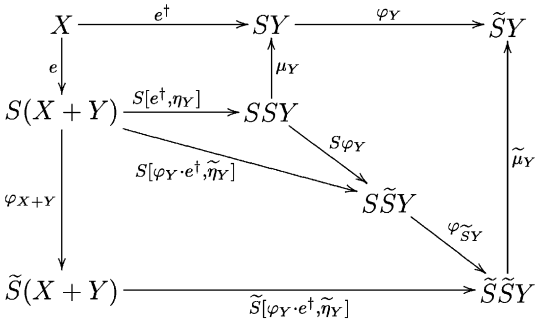
Lemma 4.11. *Monad morphisms preserve solutions of equations. That is, given a monad morphism $\varphi:\mathbb{S}\to\tilde{\mathbb{S}}$ and given an equation morphism $e:X\to S(X+Y)$ with a solution $e^\dagger:X\to SY$ (w.r.t. $\mathbb{S}$), then the equation morphism*

$$X\xrightarrow{e}S(X+Y)\xrightarrow{\varphi_{X+Y}}\tilde{S}(X+Y)$$

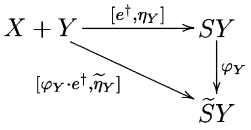
has a solution

$$X\xrightarrow{e^\dagger}SY\xrightarrow{\varphi_Y}\tilde{S}Y$$

Proof. The following diagram



commutes: for the middle triangle notice that the following triangle



commutes. □

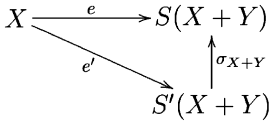
Remark 4.12.

(1) Elgot used a slightly more restrictive concept than guarded equation: his *ideal* equation morphism is an equation morphism $e : X \to S(X + Y)$ which factors through $\sigma_{X+Y} : S'(X + Y) \to S(X + Y)$. Note that all equations used in the main result, Theorem 4.14 below, are ideal, which shows that that result remains valid if complete iterativeness is defined by means of ideal, rather than guarded, equation morphisms.

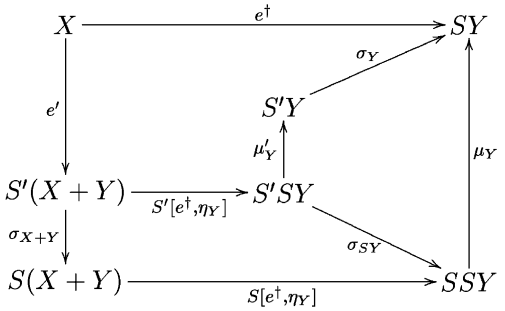
(2) Given an ideal monad $\mathbb{S}$ with $S = S' + Id$ an *ideal transformation* from a functor H to $\mathbb{S}$ is a natural transformation $H \to S$ which factors through $\sigma : S' \to S$. Example: $\tau^* : H \to T$ of Notation 3.3 is ideal.

Lemma 4.13. *For every ideal equation morphism the solution is also ideal, i.e., it factors through* σ_Y.

Proof. Given



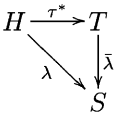
consider the following commutative diagram



□

Theorem 4.14 (**Free completely iterative monads**). *For every iteratable endofunctor H the monad $\mathbb{T}$ of Corollary* 2.18 *is a free completely iterative monad on H.*

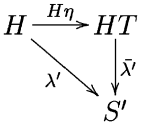
More precisely: the natural transformation $\tau^*: H \to T$ is ideal, and given a completely iterative monad $\mathbb{S} = (S, \eta^S, \mu^S)$ and an ideal transformation $\lambda: H \to S$ then there exists a unique ideal monad morphism $\bar{\lambda}: \mathbb{T} \to \mathbb{S}$ for which the following triangle



commutes.

Remark 4.15.

(1) Since $\sigma: S' \to S$, being a coproduct injection, is a (pointwise) monomorphism, the last condition on the ideal morphism $\bar{\lambda} = \bar{\lambda'} + id$ is equivalent to stating that for $\bar{\lambda'}: HT \to S'$ the following triangle



commutes.

(2) Categorically, the statement of the theorem says that every iteratable functor H in $[\mathscr{A}, \mathscr{A}]$ has a universal arrow w.r.t. the forgetful functor

$$U: \mathsf{CIM}(\mathscr{A}) \to [\mathscr{A}, \mathscr{A}]$$

of the category $\mathsf{CIM}(\mathscr{A})$ of all completely iterative monads and ideal morphisms. Beware! The functor U assigns to every completely iterative monad $\mathbb{S} = (S, \eta^S, \mu^S)$ the functor S', not S. This choice of U corresponds to the requirement that $\lambda: H \to S$ be an *ideal* transformation.

(3) The assumption that H be iteratable is fundamental: it has been proved in [23] that every endofunctor generating a free completely iterative monad is iteratable.

Proof. I. *Uniqueness of* $\bar{\lambda}$.

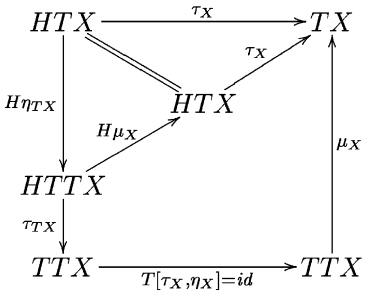
Observe that in our monad $\mathbb{T}$ the following equation morphism

$$HTX \xrightarrow{H\eta_{TX}} HTTX \xrightarrow{\tau_{TX}} TTX = T(HTX + X)$$

is guarded. Its solution is simply

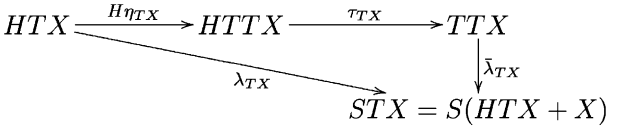
$$\tau_X : HTX \to TX.$$

In fact, the following diagram



commutes.

Suppose a monad morphism $\bar{\lambda} : \mathbb{T} \to \mathbb{S}$ as above is given. By Lemma 4.11, the following equation morphism



has the solution

$$\bar{\lambda}_X \cdot \tau_X : HTX \to SX,$$

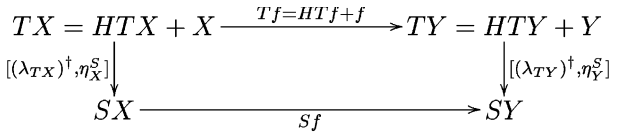
and since λ_{TX} is ideal, the solution is unique. This determines the left-hand component of $\bar{\lambda}_X : HTX + X \to SX$, and the right-hand one is clear from $\bar{\lambda}_X \cdot \eta_X = \eta^S_X$.

Shorter: we have the formula

$$\bar{\lambda}_X = [(\lambda_{TX})^\dagger, \eta^S_X]. \tag{4.1}$$

II. *Existence of* $\bar{\lambda}$. Our task is to show that, given λ, formula (4.1) defines an ideal monad morphism $\bar{\lambda} : \mathbb{T} \to \mathbb{S}$ with $\lambda = \bar{\lambda} \cdot \tau^*$.

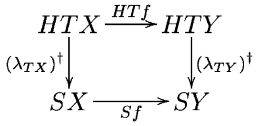
(a) Naturality of $\bar{\lambda}_X$: given a morphism $f : X \to Y$ we want to show the commutativity of the following square



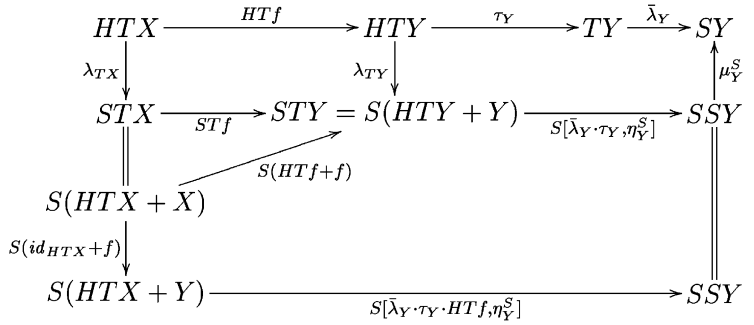
The right-hand components are clear. For the left-hand components we use the following, easily established, fact:

> Given a guarded equation morphism $e : Z \to T(Z + X)$ then also $e' = T(id + f) \cdot e : X \to T(Z + Y)$ is guarded, and $(e')^\dagger = Tf \cdot e^\dagger$, for every morphism $f : X \to Y$.

Apply this to $e = \lambda_{TX}$: we conclude that in the desired square

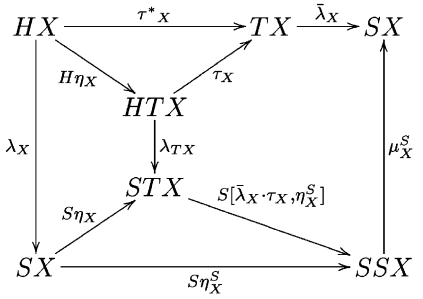


the lower passage is a solution of $e' = S(id_{HTX} + f) \cdot \lambda_{TX}$. It suffices to show that the upper passage also solves e'. This is true because the following diagram



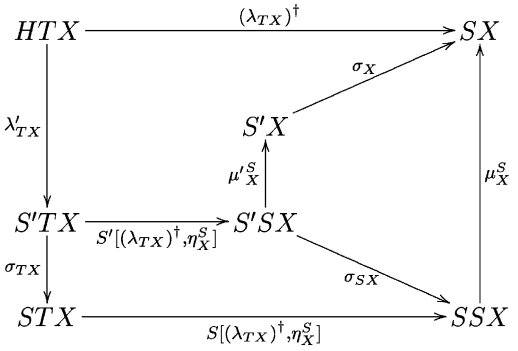
commutes. In fact, the upper right-hand square commutes due to the fact that λ_{TY} has solution $\bar{\lambda}_Y \tau_Y$, see (4.1). To see that the lower square commutes, extract S and observe that the two components obviously commute.

(b) Equality $\lambda = \bar{\lambda} \cdot \tau^*$. This follows from the next commutative diagram (where we use $\bar{\lambda}_X \cdot \tau_X = (\lambda_{TX})^\dagger$):



From $\mu^S \cdot S\eta^S = id$ we conclude that $\lambda = \bar{\lambda} \cdot \tau^*$.

(c) $\bar{\lambda}$ is an ideal monad homomorphism. In fact, since λ is an ideal transformation, say $\lambda = \sigma \cdot \lambda'$ (where λ' is unique and natural, since σ, being a coproduct injection, is pointwise monomorphic), we have for $(\lambda_{TX})^\dagger$ the following diagram



Put

$$\bar{\lambda}'_X = \mu'^S_X \cdot S'[(\lambda_{TX})^\dagger, \eta^S_{TX}] \cdot \lambda'_{TX} : HTX \to S'X$$

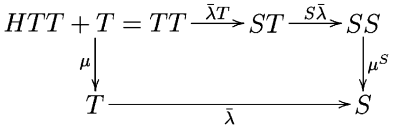
to obtain a natural transformation

$$\bar{\lambda}' : HT \to S' \quad \text{with } \bar{\lambda} = \bar{\lambda}' + id.$$

It remains to verify that $\bar{\lambda}$ is a monad morphism. Since $\eta : Id \to T$ is a coproduct injection, we have

$$\bar{\lambda}_X \cdot \eta_X = [(\lambda_{TX})^\dagger, \eta^S_X] \cdot \eta_X = \eta^S_X.$$

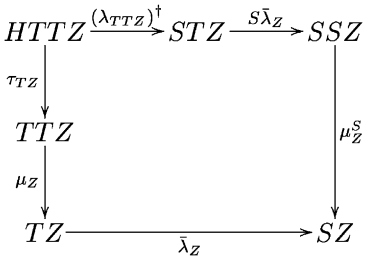
Next, we are to show that the following square



commutes. The right-hand components are both equal to $\bar{\lambda}: T \to S$: for the lower passage this follows from $\mu \cdot \eta T = id$, for the upper one from

$$(\mu^S \cdot S\bar{\lambda} \cdot \bar{\lambda}T) \cdot \eta T = \mu^S \cdot S\bar{\lambda} \cdot \eta^S T = \mu^S \cdot \eta^S S \cdot \bar{\lambda} = \bar{\lambda}.$$

Thus, we are to establish the commutativity of the left-hand components:

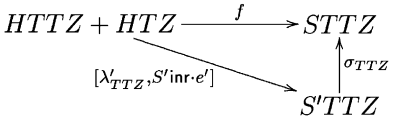


(4.2)

In the following proof of (4.2) we put $\tilde{\lambda}_Z = \lambda_{TZ}{}^\dagger : HTZ \to SZ$ and

$$f \equiv HTTZ + HTZ \xrightarrow{[\lambda_{TTZ}, S\mathsf{inr}\cdot\tilde{\lambda}_Z]} S(HTTZ + HTZ + Z) = STTZ.$$

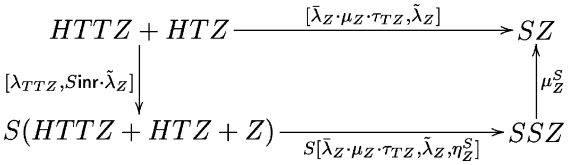
This is an equation morphism (with variables $X = HTTZ + HTZ$ and parameters Z) and it is guarded. In fact, use Lemma 4.13 on $e = \lambda_{TZ}$ to get a morphism e' with $\tilde{\lambda}_Z = \sigma_{TTZ} e'$, then the following triangle



commutes. We are going to prove that the solution of f is given as follows

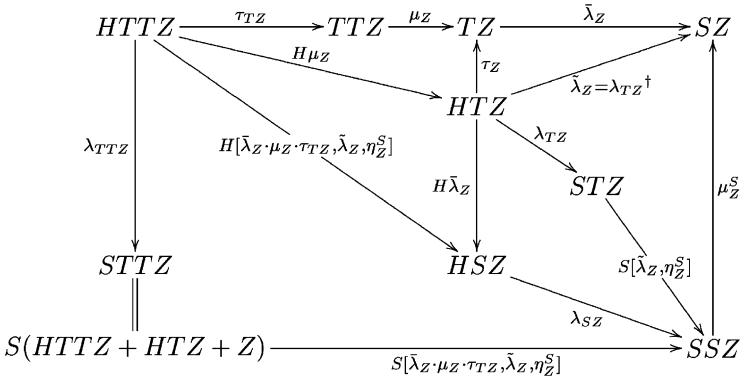
$$f^\dagger \equiv HTTZ + HTZ \xrightarrow{[\tilde{\lambda}_Z \cdot \mu_Z \cdot \tau_{TZ}, \tilde{\lambda}_Z]} SZ. \tag{4.3}$$

That is, we will verify that the following square



commutes. It is sufficient to concentrate on the left-hand components (the right-hand ones are both $\tilde{\lambda}_Z$ due to $\mu^S_Z \cdot S\eta^S_Z = id$). For this we consider the following

diagram:

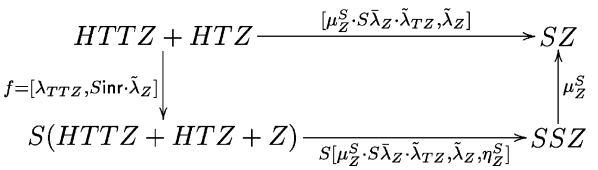


All parts commute: this is obvious, except for the middle triangle. We show that this commutes even if we delete H. Use $TTZ = HTTZ + HTZ + Z$ with coproduct injections τ_{TZ}, $\eta_{TZ} \cdot \tau_Z$ and $\eta_{TZ} \cdot \eta_Z$ respectively: the left-hand components are $\bar{\lambda}_Z \cdot \mu_Z \cdot \tau_{TZ}$, the middle ones are $\tilde{\lambda}_Z = \bar{\lambda}_Z \cdot \mu_Z \cdot \eta_{TZ} \cdot \tau_Z = \bar{\lambda}_Z \cdot \tau_Z$, and the right-hand ones are $\eta_Z^S = \bar{\lambda}_Z \cdot \eta_Z$. This proves (4.3).

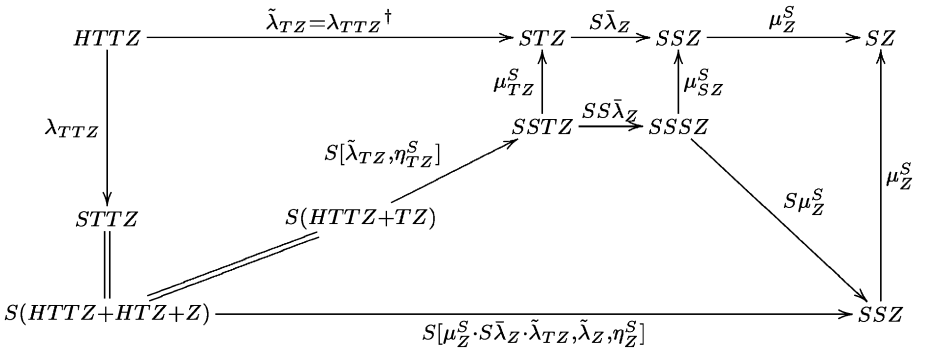
But the morphism f also has the following solution

$$f^\dagger \equiv HTTZ + HTZ \xrightarrow{[\mu_Z^S \cdot S\tilde{\lambda}_Z \cdot \tilde{\lambda}_{TZ}, \tilde{\lambda}_Z]} SZ. \tag{4.4}$$

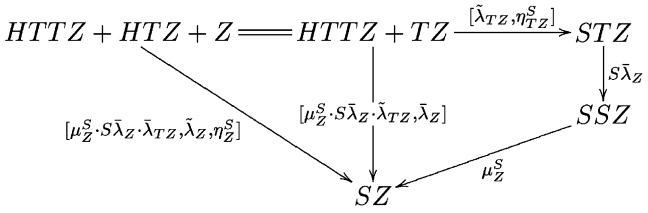
In fact, the following square



commutes: the right-hand components commute trivially (as above) and for the left-hand ones consider the following diagram:



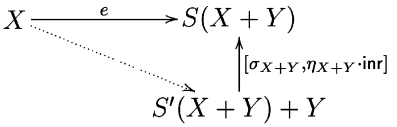
It commutes: this is obvious for all parts except the lower part, for which we delete S to obtain



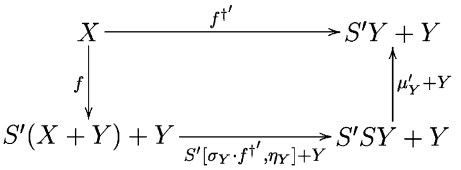
which commutes since $\mu^S \cdot \eta^S S = id$.

Since solutions are unique, the two solutions of f above are equal. The equality of the right-hand components in (4.3) and (4.4) is precisely the fact that (4.2) above commutes. This concludes the proof of (c). □

Remark 4.16. The above theorem holds, more generally, in categories $\mathscr{A}$ with binary coproducts also when we do not assume that coproduct injections are monomorphisms. However, we have to define ideal equations and solutions differently, then. In the present approach, a guarded equation morphism $e : X \to S(X + Y)$ is one that factors as



and, as long as coproduct injections are monomorphisms, we do not need a name for the factorizing arrow. Now generally, we can introduce guarded equation morphisms as arrows $f : X \to S'(X + Y) + Y$. And a solution of f is, then, defined as a morphism $f^{\dagger'} : X \to S'Y + Y$ such that the following diagram



commutes. An ideal monad $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ is called completely iterative if every guarded equation arrow f has a unique solution $f^{\dagger'}$.

In this greater generality it remains true that for every iteratable functor H

(i) the monad $\mathbb{T}$ is completely iterative,

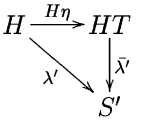
and

(ii) $\mathbb{T}$ is a free completely iterative monad on H.

The latter means, now, that for every completely iterative monad $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ and every natural transformation $\lambda' : H \to S'$ there exists a unique monad morphism

$$\bar{\lambda} : \mathbb{T} \to \mathbb{S}$$

such that
(a) $\bar{\lambda}$ is ideal, i.e., has the form $\bar{\lambda} = \bar{\lambda}' + id$ for $\bar{\lambda}' : HT \to S'$,
and
(b) the triangle of Remark 4.15



commutes.
In other words, the functor U of Remark 4.15 has a universal arrow for every iteratable H. The proof is the same as the proof of Theorem 4.14 above.

5. A completely iterative monoid of an object

We can view the procedure of forming the monad $\mathbb{T}$ of Section 2 globally by working, instead of in the given category $\mathscr{A}$, in the endofunctor category $[\mathscr{A}, \mathscr{A}]$. Here H is an object. If H is iteratable, then 2.21 defines another object, T, together with a morphism (natural transformation)

$$\alpha : T \to HT + Id.$$

This is a coalgebra of the functor

$$\hat{H} : [\mathscr{A}, \mathscr{A}] \to [\mathscr{A}, \mathscr{A}]$$

defined on objects by

$$\hat{H}(S) = H \cdot S + Id \quad (\text{for all } S : \mathscr{A} \to \mathscr{A})$$

and analogously on morphisms. We prove below that T is a final $\hat{H}$-coalgebra.

Within the realm of locally small categories (i.e., with small hom-sets) with coproducts this global approach is equivalent to that of Section 2:

Proposition 5.1. *Let $\mathscr{A}$ be a locally small category with coproducts. For every endofunctor H, the following are equivalent:*
(1) *H is an iteratable object of $[\mathscr{A}, \mathscr{A}]$, i.e., a final $\hat{H}$-coalgebra exists.*
(2) *H is an iteratable endofunctor, i.e., all final $(H(\_) + X)$-coalgebras exist.*

Remark.
(i) More detailed: if T is a final $\hat{H}$-coalgebra, we prove that TX is a final coalgebra of $H(\_) + X$ for all objects X. And vice versa.
(ii) The proof that 2 implies 1 holds for all categories $\mathscr{A}$ with binary coproducts.

For the proof that 1 implies 2, only copowers indexed by hom-sets of the category $\mathscr{A}$ are used. Thus the proposition also holds e.g. for the category $\mathscr{A} = \mathsf{Set}_{fin}$ of finite sets, and for any poset $\mathscr{A}$ with binary joins.

Proof. 1 implies 2: For every pair X, Y of objects in $\mathscr{A}$ denote by $K_{X,Y}$ the following endofunctor

$$K_{X,Y}A = \coprod_{\mathscr{A}(X,A)} Y$$

for objects A, analogously for morphisms. This is just a left Kan extension of Y, considered as a functor $\mathsf{1} \to \mathscr{A}$, along the functor $X : \mathsf{1} \to \mathscr{A}$. In fact, for every functor $P : \mathscr{A} \to \mathscr{A}$ we have a bijection

$$\frac{K_{X,Y} \to P}{Y \to PX}$$

natural in P, which to every natural transformation $\varphi : K_{X,Y} \to P$ assigns the composite

$$Y \xrightarrow{u} \coprod_{\mathscr{A}(X,X)} Y \xrightarrow{\varphi_X} PX,$$

where u is the id_X-injection. Conversely, given a morphism $f : Y \to PX$, the corresponding natural transformation $f^{@} : K_{X,Y} \to P$ has components

$$f^{@}_{A} : \left(\coprod_{h : X \to A} Y \right) \to PA$$

determined by $Y \xrightarrow{f} PX \xrightarrow{Ph} PA$.

Let $\alpha : T \to HT + Id$ be a final $\hat{H}$-coalgebra. We will show that

$$\alpha_X : TX \to HTX + X$$

is a final $(H(\_) + X)$-coalgebra for every X.

In fact, for every $(H(\_) + X)$-coalgebra

$$b : Y \to HY + X$$

when composing b with

$$Hu + id : HY + X \to H\left(\coprod_{\mathscr{A}(X,X)} Y \right) + X = (\hat{H}K_{X,Y})X$$

we obtain a morphism

$$\bar{b} : Y \to (\hat{H}K_{X,Y})X$$

which by the above adjointness yields an $\hat{H}$-coalgebra

$$\bar{b}^{@} : K_{X,Y} \to \hat{H}K_{X,Y}.$$

Let φ be the unique homomorphism of $\hat{H}$-coalgebras

$$\begin{array}{ccc} K_{X,Y} & \xrightarrow{\bar{b}^{@}} & \widehat{H}K_{X,Y} \\ {\scriptstyle\varphi}\downarrow & & \downarrow{\scriptstyle\widehat{H}\varphi} \\ T & \xrightarrow[\alpha]{} & \widehat{H}T \end{array}$$

Then $\varphi = f^{@}$ for a unique $f : Y \to TX$, and the commutativity of the above square yields the commutativity of

$$\begin{array}{ccc} Y & \xrightarrow{b} & HY + X \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle Hf+id_X} \\ TX & \xrightarrow[\alpha_X]{} & HTX + X \end{array}$$

2 implies 1: It has been noted above (see Corollary 2.18) that if $\alpha_X : TX \to HTX + X$ denotes a final coalgebra for $H(\_) + X$, then the assignment $X \mapsto TX$ can be extended to a functor $T : \mathscr{A} \to \mathscr{A}$.

Analogously one can show that the collection of all α_X's constitutes a natural transformation $\alpha : T \to H \cdot T + Id$. Thus, α makes T an $\hat{H}$-coalgebra.

To verify that α is indeed a final $\hat{H}$-coalgebra, consider any coalgebra $\beta : S \to H \cdot S + Id$. For each X in $\mathscr{A}$ there exists a unique morphism $f_X : SX \to TX$ such that the following square

$$\begin{array}{ccc} SX & \xrightarrow{\beta_X} & HSX + X \\ {\scriptstyle f_X}\downarrow & & \downarrow{\scriptstyle Hf_X+id} \\ TX & \xrightarrow[\alpha_X]{} & HTX + X \end{array}$$

commutes. It is easy to show that the collection of f_X's is natural in X and that it defines a unique natural transformation $f : S \to T$ for which the following square

$$\begin{array}{ccc} S & \xrightarrow{\beta} & HS + Id \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle Hf+id} \\ T & \xrightarrow[\alpha]{} & HT + Id \end{array}$$

commutes. □

Remark 5.2. In Example 2.15 we have formulated properties of a category $\mathscr{A}$ so that every continuous endofunctor H be iteratable. Let us observe that the corresponding completely iterative monad, T, is also continuous: by Proposition 5.1, T is a final $\hat{H}$-coalgebra. Now $\hat{H}$ is an endofunctor of the category $[\mathscr{A}, \mathscr{A}]$ which also satisfies 1.–3, of Example 2.15. Consequently, we have the formula

$$T = \lim_{n<\omega} \hat{H}^n(C_1),$$

where C_1 (the constant endofunctor of $\mathscr{A}$ with value 1) is a terminal object of $[\mathscr{A}, \mathscr{A}]$. Since each $\hat{H}(C_1)$ is easily seen to be continuous, we obtain T as a limit of continuous functors—thus, T is continuous.

Remark 5.3. For every category $\mathscr{A}$ the endofunctor category $[\mathscr{A}, \mathscr{A}]$ is monoidal with composition as a tensor product and Id as a unit. Moreover composition distributes

over coproducts on the left: $(H+K)\cdot L=(H\cdot L)+(K\cdot L)$. This leads us to consider an arbitrary monoidal category

$$(\mathscr{B},\otimes,I)$$

with coherence isomorphisms (for all H, K, L in $\mathscr{B}$):

$$l_H : I\otimes H\to H,\quad r_H : H\otimes I\to H$$

and

$$a_{H,K,L} : H\otimes(K\otimes L)\to(H\otimes K)\otimes L$$

satisfying the usual laws, and which is left-distributive in the following sense:

Definition 5.4.

(1) A monoidal category is called *left-distributive* if it has binary coproducts and the canonical morphisms

$$d_{H,K,L} : (H\otimes L)+(K\otimes L)\to(H+K)\otimes L$$

are all isomorphisms.

(2) An object H of a monoidal category $\mathscr{B}$ is said to be *iteratable* provided that the endofunctor $\hat{H} : \mathscr{B}\to\mathscr{B}$ defined by

$$\hat{H}(B)=H\otimes B+I$$

has a final coalgebra.

(3) A left distributive monoidal category with each object iteratable is called an *iteratable category*.

Examples 5.5.

(1) The category

$$Cont[\mathsf{Set},\mathsf{Set}]$$

of continuous endofunctors (i.e., those preserving ω^{op}-limits) of Set is iteratable: we know that continuous functors are closed under
(a) composition (here: a tensor product)
(b) identity functor (here: unit I)
and
(c) finite coproducts,
thus $Cont[\mathsf{Set},\mathsf{Set}]$ is a distributive monoidal subcategory of $[\mathsf{Set},\mathsf{Set}]$. Now, every continuous functor is iteratable, and by Remark 5.2 the completely iterative monad is also continuous; therefore $Cont[\mathsf{Set},\mathsf{Set}]$ is an iteratable category.

(2) More in general, $Cont[\mathscr{A},\mathscr{A}]$ is an iteratable category for every locally small category $\mathscr{A}$ satisfying conditions 1.–3, of Example 2.15.

(3) The category

$$Fin[\mathsf{Set},\mathsf{Set}]$$

of all finitary endofunctors of Set (i.e., those preserving filtered colimits) is iteratable. In fact, finitary functors are closed under composition, identity functor, and finite coproducts, thus, $Fin[\mathsf{Set},\mathsf{Set}]$ is a distributive monoidal subcategory of $[\mathsf{Set},\mathsf{Set}]$.

A completely iterative monad $\mathbb{T}$ of a finitary functor H exists, since finitary functors always have final coalgebras, see [11], Theorem 1.2, and each $H(\_)+X$ is clearly finitary. However, this monad is seldom finitary, see Example 2.22(2).

We can form a finitary part $\mathbb{T}_{fin}$ of every monad $\mathbb{T}$ on Set (see [21]): it is obtained by restricting the underlying functor T to the full subcategory Set_{fin} of finite sets, and then forming a left Kan extension of T/Set_{fin} along the embedding of Set_{fin} in Set.

It is easy to verify that $\mathbb{T}_{fin}$ is a final coalgebra of the endofunctor $H\cdot(\_)+Id$ of $Fin[\mathsf{Set},\mathsf{Set}]$. In fact, given any coalgebra

$$S \to H\cdot S + Id$$

(with S finitary, of course) the unique $\hat{H}$-homomorphism $f: S \to T$ is easily seen to have a factorization through the canonical morphism $m: T_{fin} \to T$. That is, we have a unique $f': S \to T_{fin}$ with $f = m\cdot f'$. And f' is the unique homomorphism of coalgebras of the functor $H\cdot(\_)+Id$, considered as an endofunctor of $Fin[\mathsf{Set},\mathsf{Set}]$.

Example: the functor

$$H : \mathsf{Set} \to \mathsf{Set} \quad \text{with } HZ = Z \times Z$$

has the completely iterative monad $\mathbb{T}$ where TX are all binary trees with leaves indexed in X. And $\mathbb{T}_{fin}$ is the finitary monad where $T_{fin}X$ are all binary trees with leaves indexed in a finite subset of X.

(4) More generally, if $\mathscr{A}$ is a locally finitely presentable category (see [8]) then $Fin[\mathscr{A},\mathscr{A}]$, the category of finitary endofunctors of $\mathscr{A}$, is iteratable. The argument is the same: we form a completely iterative monad $\mathbb{T}$ in $[\mathscr{A},\mathscr{A}]$, which exists by Theorem 1.2 in [11] (although formulated for Set, it holds in all locally presentable categories) and then take a finitary part $\mathbb{T}_{fin}$ just as in (3) above.

(5) Let $\mathscr{B}$ be a left distributive monoidal category having a terminal object 1 and limits of ω^{op}-chains which commute with both the tensor product and the binary coproduct. Then every object H is iteratable and T is a limit of the following countable chain:

$$1 \xleftarrow{!} H\otimes 1 + I \xleftarrow{H\otimes !+id} H\otimes(H\otimes 1 + I) + I \xleftarrow{H\otimes(H\otimes !+id)+id} \cdots$$

For example: the category of sets with a binary product as $\otimes$ and a terminal object I as a unit is an iteratable category: the (polynomial) functor

$$\hat{H}(Z) = H\times Z + I$$

has a final coalgebra

$$T = H^{\infty}$$

for every set H.

And the cartesian closed category Cat of all small categories is an iteratable category. Every small category H is iterable with

$$T = 1 + H + (H \times H) + \cdots + H^{\omega}$$

(6) Let H be an iteratable Abelian group (where we consider the category Ab of all Abelian groups with the usual tensor product). Then a final coalgebra of $\hat{H}$ is, as we show below in 5.8, a monoid in the given monoidal category—thus, in the present case

T is a ring.

Notation 5.6. For every iteratable object H we denote by T and $\alpha : T \to H \otimes T + I$ a final coalgebra of $\hat{H}$. By Lambek's Lemma, T is a coproduct of $H \otimes T$ and I. We denote the injections by

$$\tau : H \otimes T \to T \quad \text{and} \quad \eta : I \to T$$

where $\alpha^{-1} = [\tau, \eta]$.

This makes T into an algebra for the functor $H \otimes \_$. More generally, every object S of $\mathscr{B}$ yields an algebra

$$\tau_S \equiv H \otimes (T \otimes S) \xrightarrow{a_{H,T,S}} (H \otimes T) \otimes S \xrightarrow{\tau \otimes id_S} T \otimes S$$

(where $a_{H,T,S}$ is the associativity isomorphism). Put

$$\eta_S \equiv S \xrightarrow{r_S} I \otimes S \xrightarrow{\eta \otimes id_S} T \otimes S.$$

Substitution Theorem 5.7. *Let H be an iteratable object in a monoidal category $\mathscr{B}$. For every morphism*

$$s : S \to T$$

in $\mathscr{B}$ there is a unique homomorphism

$$\hat{s} : T \otimes S \to T$$

of algebras of type $H \otimes \_$ with

$$s = \hat{s} \cdot \eta_S.$$

Proof. This is quite analogous to the proof of Theorem 2.17. We turn the object $T \otimes S + T$ into an $\hat{H}$-coalgebra as follows:

$$T \otimes S + T \cong H \otimes T \otimes S + S + T \xrightarrow{id + [s,id]} H \otimes T \otimes S + T \cong$$

$$\cong H \otimes T \otimes S + H \otimes T + I \xrightarrow{[H \otimes id, H \otimes \mathsf{inr}] + id} H \otimes (T \otimes S + T) + I.$$

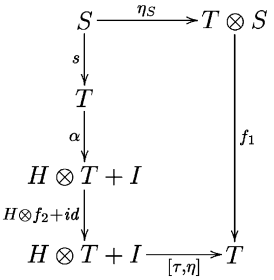
The unique homomorphism

$$f = [f_1, f_2] : T \otimes S + T \to T$$

of $\hat{H}$-coalgebras is the unique morphism of $\mathscr{B}$ which has the second component, f_2, an endomorphism of the final $\hat{H}$-coalgebra $\alpha: T \to H \otimes T + I$, thus,

$$f_2 = id,$$

and for the first component we get two commutative diagrams: one tells us that f_1 is a homomorphism of $(H \otimes \_)$-algebras, and the other one is as follows:



Since $f_2 = id$, this diagram tells us that $f_1 \cdot \eta_S = s$, which proves the Substitution Theorem. □

Corollary 5.8. *For every iteratable object H, a final $\hat{H}$-coalgebra T is a monoid with respect to*

$$\eta: I \to T$$

and

$$\mu = \widehat{id_T}: T \otimes T \to I.$$

Proof. In fact, the equality $\mu \cdot \eta_T = id$ follows from the definition of μ and the other two equalities defining monoids in $(\mathscr{B}, \otimes, I)$ easily follow from the uniqueness of $\hat{s}$. □

Definition 5.9. The monoid of the above corollary is called a *completely iterative monoid generated by* an iteratable object H.

We now prove a remarkable property of iteratable categories $\mathscr{B}$: denote by

$$\mathscr{T}: \mathscr{B} \to \mathscr{B}$$

the functor assigning to every object H a completely iterative monoid generated by H. Then $\mathscr{T}$, as an object of $[\mathscr{B}, \mathscr{B}]$, is itself a completely iterative monoid: it is generated by $Id_{\mathscr{B}}$. Example: Set is an iteratable category, see Example 5.5(5), and the assignment $H \mapsto H^\infty$ is, as an object of [Set, Set], itself a completely iterative monoid generated by Id.

For every monoidal category $\mathscr{B}$ we consider $[\mathscr{B}, \mathscr{B}]$ as a monoidal category (with the "pointwise" tensor product $P \otimes Q: H \mapsto P(H) \otimes Q(H)$ and the "pointwise" unit $C_I: H \mapsto I$).

Theorem 5.10. *Suppose that* $(\mathscr{B},\otimes,I)$ *is an iteratable category. Then the following hold:*

(1) *The functor category* $[\mathscr{B},\mathscr{B}]$ *is iteratable.*

(2) *The assignment of a completely iterative monoid to every object is an endofunctor of* $\mathscr{B}$ *which, as an object of* $[\mathscr{B},\mathscr{B}]$, *is itself a completely iterative monoid generated by* $Id_{\mathscr{B}}$.

Proof. 1. First observe that $[\mathscr{B},\mathscr{B}]$ is indeed a distributive monoidal category, since the required structure is transported pointwise from $\mathscr{B}$.

Consider now any functor $H:\mathscr{B}\to\mathscr{B}$. To show that the derived functor

$$\hat{H} = H\otimes(\_)+C_I : [\mathscr{B},\mathscr{B}]\to[\mathscr{B},\mathscr{B}]$$

has a final coalgebra, form, for each B in $\mathscr{B}$, a final coalgebra of the functor $H(B)\otimes(\_)+I$:

$$a_B : T(B)\to H(B)\otimes T(B)+I.$$

It is clear that there is a unique canonical way of making the assignment $B\mapsto T(B)$ functorial: consider any morphism $f:B\to C$ in $\mathscr{B}$ and define $T(f):T(B)\to T(C)$ to be the unique morphism such that the following diagram

$$\begin{array}{ccccc} T(B) & \xrightarrow{a_B} & H(B)\otimes T(B)+I & \xrightarrow{H(f)\otimes T(B)+id} & H(C)\otimes T(B)+I \\ {\scriptstyle T(f)}\downarrow & & & & \downarrow{\scriptstyle H(C)\otimes T(f)+id} \\ T(C) & & \xrightarrow[a_C]{} & & H(C)\otimes T(C)+I \end{array}$$

commutes. It is easy to show that this indeed defines a functor $T:\mathscr{B}\to\mathscr{B}$.

The collection of morphisms $a_B:T(B)\to H(B)\otimes T(B)+I$ is natural in B and thus defines a coalgebra for $H\otimes(\_)+C_I$:

$$a : T\to H\otimes T+C_I.$$

To show that a is a final coalgebra, consider any coalgebra

$$b : S\to H\otimes S+C_I.$$

For every B in $\mathscr{B}$ there exists a unique morphism $\lambda_B:S(B)\to T(B)$ such that the following square

$$\begin{array}{ccc} S(B) & \xrightarrow{b_B} & H(B)\otimes S(B)+I \\ {\scriptstyle \lambda_B}\downarrow & & \downarrow{\scriptstyle H(B)\otimes\lambda_B+id} \\ T(B) & \xrightarrow[a_B]{} & H(B)\otimes T(B)+I \end{array}$$

commutes. To show that the collection (λ_B) constitutes a natural transformation, observe that, for every $f : B \to C$, both

$$\lambda_C \cdot S(f) : S(B) \to T(C) \quad \text{and} \quad T(f) \cdot \lambda_B : S(B) \to T(C)$$

are homomorphisms of $(H(C) \otimes (\_) + I)$-coalgebras from

$$(H(f) \otimes S(B) + id) \cdot b_B : S(B) \to H(C) \otimes S(B) + I$$

to

$$a_C : T(C) \to H(C) \otimes T(C) + I$$

and therefore they are equal.

We have formed a final coalgebra

$$a : T \to H \otimes T + C_I.$$

2. Put $\Phi(B) = T_B$ for every object B, where T_B denotes a completely iterative monoid generated by B, and extend the assignment $B \mapsto \Phi(B)$ to a functor $\Phi : \mathscr{B} \to \mathscr{B}$ as in the first part of the proof.

Let us now consider the functor

$$Id \otimes (\_) + C_I : [\mathscr{B}, \mathscr{B}] \to [\mathscr{B}, \mathscr{B}].$$

The collection of morphisms $a_B : \Phi(B) \to B \otimes \Phi(B) + I$ defines a coalgebra for $Id \otimes (\_) + C_I$:

$$a : \Phi \to Id \otimes \Phi + C_I$$

and it follows from the first part of the proof that this coalgebra is final.

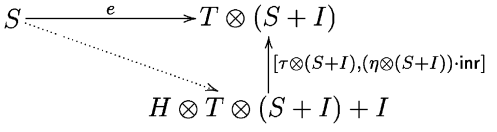
To conclude the proof use the monoidal version of the existence of a completely iterative monad from Corollary 5.8. □

Finally, we show that if H is an iteratable object (with the corresponding monoid T) of a left distributive monoidal category $\mathscr{B}$, then guarded equation morphisms have unique solutions.

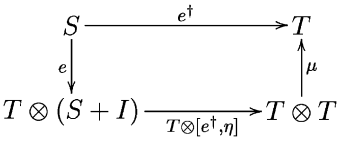
Definition 5.11. Let H be an iteratable object of a left distributive category $\mathscr{B}$ with a completely iterative monoid T. Every morphism of the form

$$e : S \to T \otimes (S + I) \qquad S \text{ an object of } \mathscr{B}$$

is called an *equation morphism*. It is called *guarded* if it factors through $[\tau \otimes (S + I), (\eta \otimes (S + I)) \cdot \mathsf{inr}]$:



Solution Theorem 5.12. *For every iteratable object H every guarded equation morphism $e : S \to T \otimes (S + I)$ has a unique solution, i.e., there exists a unique morphism $e^\dagger : S \to T$ such that the following diagram*



commutes.

Proof. The proof is analogous to the proof of Corollary 3.8. □

References

[1] P. Aczel, Non-well-founded Sets, CSLI Lecture Notes No. 14, Stanford University, Stanford, CA, 1988.
[2] P. Aczel, J. Adámek, J. Velebil, A coalgebraic view of infinite trees and iteration, Electron. Notes Theoret. Comput. Sci. 44 (1) (2001).
[3] J. Adámek, Free algebras and automata realizations in the language of categories, Comment. Math. Univ. Carolin. 15 (1974) 589–602.
[4] J. Adámek, Final coalgebras are cauchy completions of initial algebras. Theoret. Comput. Sci., to appear.
[5] J. Adámek, V. Koubek, On the greatest fixed point of a set functor, Theoret. Comput. Sci. 150 (1995) 57–75.
[6] J. Adámek, H.-E. Porst, From varieties of algebras to covarieties of coalgebras, Electron. Notes Theoret. Comput. Sci. 44 (1) (2001).
[7] J. Adámek, J. Reiterman, Banach's fixed-point theorem as a base for data-type equations, Appl. Categorical Struct. 2 (1994) 77–90.
[8] J. Adámek, J. Rosický, Locally Presentable and Accessible Categories, Cambridge University Press, Cambridge, 1994.
[9] J. Adámek, V. Trnková, Automata and Algebras in Categories, Kluwer Academic Publishers, Dordrecht, 1990.
[10] P. America, J. Rutten, Solving reflexive domain equations in a category of complete metric spaces, J. Comput. System Sci. 39 (1989) 343–375.
[11] M. Barr, Terminal coalgebras in well-founded set theory, Theoret. Comput. Sci. 114 (1993) 299–315.
[12] J. Barwise, J. Etchemendy, The Liar, Oxford University Press, Oxford, 1987.
[13] J. Barwise, L. Moss, Vicious Circles, CSLI Lecture Notes No. 60, Stanford University, Stanford, CA, 1996.
[14] S.L. Bloom, Z. Ésik, Iterative Theories: The Equational Logic of Iterative Processes, EATCS Monograph Series on Theoretical Computer Science, Springer, Berlin, 1993.
[15] C.C. Elgot, Monadic computation and iterative algebraic theories, in: H.E. Rose, J.C. Shepherdson (Eds.), Logic Colloquium '73, North-Holland Publishers, Amsterdam, 1975.
[16] C.C. Elgot, S.L. Bloom, R. Tindell, On the algebraic structure of rooted trees, J. Comput. System Sci. 16 (1978) 361–399.
[17] N. Ghani, C. Lüth, F. de Marchi, J. Power, Algebras, coalgebras, monads and comonads, Electron. Notes Theoret. Comput. Sci. 44 (1) (2001).
[18] J.A. Goguen, S.W. Thatcher, E.G. Wagner, J.B. Wright, Initial algebra semantics and continuous algebras, J. ACM 24 (1977) 68–95.
[19] T. Jech, Set Theory, Academic Press, New York, 1978.
[20] J. Lambek, A fixpoint theorem for complete categories, Math. Z. 103 (1968) 151–161.
[21] F.E.J. Linton, Some aspects of equational categories, in: Proc. Conf. Categorical Algebra, La Jolla, 1965, Springer, Berlin, 1966, 84–94.

[22] M. Makkai, R. Paré, Accessible Categories: The Foundations of Categorical Model Theory, in: Contemporary Mathematics, Vol. 104, AMS, Providence, RI, 1989.
[23] S. Milius, On iteratable endofunctors, Electron. Notes Theoret. Comput. Sci. 69 (2002).
[24] L. Moss, Parametric corecursion, Theoret. Comput. Sci. 260 (2001) 139–163.
[25] L. Moss, Recursion and corecursion have the same equational logic, preprint, available at http://math.indiana.edu/home/moss/eqcoeq.ps
[26] M.B. Smyth, G.D. Plotkin, The category-theoretic solution of recursive domain equations, SIAM J. Comput. 11 (1982) 761–783.
[27] J. Tiuryn, Unique fixed points vs. least fixed points, Theoret. Comput. Sci. 12 (1980) 229–254.

Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information and Computation 196 (2005) 1–41
Information and Computation
www.elsevier.com/locate/ic

Completely iterative algebras and completely iterative monads

Stefan Milius

Institute of Theoretical Computer Science, Technical University, Braunschweig, Germany

Received 27 September 2003

Abstract

Completely iterative theories of Calvin Elgot formalize (potentially infinite) computations as solutions of recursive equations. One of the main results of Elgot and his coauthors is that infinite trees form a free completely iterative theory. Their algebraic proof of this result is extremely complicated. We present completely iterative algebras as a new approach to the description of free completely iterative theories. Examples of completely iterative algebras include algebras on complete metric spaces. It is shown that a functor admits an initial completely iterative algebra iff it has a final coalgebra. The monad given by free completely iterative algebras is proved to be the free completely iterative monad on the given endofunctor. This simplifies substantially all previous descriptions of these monads. Moreover, the new approach is much more general than the classical one of Elgot et al. A necessary and sufficient condition for the existence of a free completely iterative monad is proved.

Keywords: Completely iterative algebra; Coalgebra; Completely iterative theory; Monad

1. Introduction

The goal of the current paper is the study of completely iterative algebras (cia), i.e., algebras in which every system of recursive equations has a unique solution. This study allows a new approach to completely iterative theories, which were introduced and studied by Elgot et al. [10]. Completely iterative theories allow the treatment of the semantics of potentially infinite computations of a

Email address: milius@iti.cs.tu-bs.de.

0890-5401/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.ic.2004.05.003

computer program in an algebraic setting abstracting away from the nature of the external memory. They are algebraic theories (in the sense of Lawvere [13] and Linton [14]) that allow for unique solutions of fixed point equations. An important example of a completely iterative theory is the theory of finite and infinite trees over a signature Σ. In [10] it is shown that this is the free completely iterative theory over Σ.

In recent years it has been realized that a more abstract categorical approach to completely iterative theories allows to generalize the classical results beyond the universal algebra setting. Moreover, the proofs become substantially simpler and conceptually much clearer, see the work of Moss [17] and the work of Aczel et al. [1]. To be a bit more precise, in lieu of a signature one starts with an endofunctor H on Set (or more generally, any category $\mathcal{A}$ with binary coproducts) having "enough final coalgebras," i.e., for any object Y there exists a final coalgebra TY of $H(\_) + Y$. The main result of [1] is that T is a free completely iterative monad on H.

In the present paper, we add completely iterative algebra to the picture, and we establish for every category $\mathcal{A}$ with binary coproducts, and every endofunctor H on $\mathcal{A}$ that given an object mapping T of $\mathcal{A}$ the following three statements are equivalent:

(a) for every object Y, TY is a final coalgebra of $H(\_) + Y$,
(b) for every object Y, TY is a free completely iterative H-algebra on Y, and
(c) T is a free completely iterative monad on H.

The implication that (a) implies (c) is the main result of [1]. The converse (c) implies (a) is a new result. It has appeared before in the extended abstract [16] but not in a journal article. The main contribution of the current paper is to add (b) to the above list. Here we shall first establish the equivalence of (a) and (b), and then we prove that (b) implies (c). This leads to a substantial simplification of the proof of [1]. For the converse (c) implies (b) we use the technical material from [16], and we take here the opportunity to streamline it a bit. More on the technical side this material will allow us to drop an annoying little side condition of our results in [1]—there coproduct injections were assumed to be monomorphic—and the freeness in (c) can be slightly extended.

In Section 1, we shall restrict ourselves to the classical case to clarify our results a bit more. So suppose we are given a *polynomial* endofunctor H_Σ on the category Set, i.e., one that is obtained from a signature $\Sigma = (\Sigma_n)_{n<\omega}$ as follows:

$$H_\Sigma X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X^2 + \cdots.$$

Thus, the classical Σ-algebras are precisely the algebras of the functor H_Σ. From Section 2 on we shall work more generally with an endofunctor on an arbitrary category with binary coproducts.

A Σ-algebra A is called *completely iterative*, if every system

$$x_i \approx t_i, \quad i \in I, \tag{1.1}$$

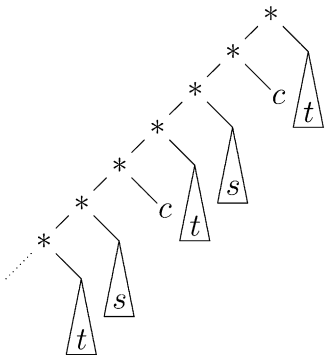
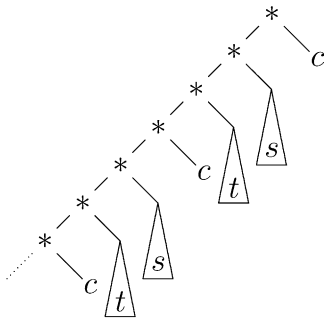
where I is some (possibly infinite) set, $X = \{ x_i \mid i \in I \}$ is a set of variables and the t_i are terms over $X + A$, none of which is just a single variable, has a unique solution in A. By a *solution* we mean a set $\{ x_i{}^\dagger \mid i \in I \}$ of elements of A such that the above formal equations (1.1) become actual identities in A when the variables are substituted by the solutions and the terms t_i are interpreted in A, i.e.,

$$x_i{}^\dagger \equiv t_i\left(\{ x_j{}^\dagger / x_j \mid j \in I \}\right), \quad i \in I.$$

Example. Suppose we have a signature Σ. The algebra $A = T_\Sigma$ of all finite and infinite Σ-trees, i.e., trees whose nodes with n children are labelled by n-ary operation symbols from Σ, is completely iterative. For example, let Σ consist of a binary operation symbol $*$ and a constant symbol c. Then the following system:

$$x_1 \approx x_2 * t \quad x_2 \approx (x_1 * s) * c, \tag{1.2}$$

where s and t are some trees in T_Σ has the following solution:

$x_1{}^\dagger =$  and $x_2{}^\dagger =$ 

Observe that it is sufficient to allow for the right-hand side in (1.2) only so-called *flat terms*, i.e., terms t that are either

$$t = \sigma(x_1, \ldots, x_n), \qquad \sigma \in \Sigma_n, \quad x_1, \ldots, x_n \in X,$$

or

$$t \in A.$$

In fact, for every system (1.1) one can give a system with only flat terms on the right-hand side, which has the same solution. This is done by introducing (possibly infinitely many) new variables. For example for the system (1.2) we get the following flat one:

$$\begin{array}{ll} x_1 \approx x_2 * z_1 & z_2 \approx x_1 * z_4 \\ x_2 \approx z_2 * z_3 & z_3 \approx c \\ z_1 \approx t & z_4 \approx s \end{array}$$

Obviously, the solutions $x_1{}^\dagger$ and $x_2{}^\dagger$ are the same trees as before.

Clearly, one can write every system with flat right-hand sides as a single map

$$e : X \longrightarrow H_\Sigma X + A$$

and a solution is a map $e^\dagger : X \longrightarrow A$ such that the following square

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ H_\Sigma X + A & \xrightarrow[H_\Sigma e^\dagger + A]{} & H_\Sigma A + A \end{array}$$

where a denotes the algebra structure of A, commutes. We call an algebra A *completely iterative* if any flat equation morphism e has a unique solution $e^\dagger$. Among classical algebras the property of being completely iterative seems to be quite rare. However, there exist interesting examples of completely iterative algebras, e.g., the algebras

$$T_\Sigma$$

of finite and infinite Σ-trees form a completely iterative algebra, in fact, we prove below that T_Σ is the initial completely iterative Σ-algebra. It follows that for any set Y the algebra

$$T_\Sigma Y$$

of all finite and infinite Σ-trees with leaves labelled by constant symbols from Σ or variables from Y is a free cia on Y. The free cias define a monad $\mathbb{T}_\Sigma$ on Set, and this monad is the free completely iterative monad on H_Σ.

In our proof we work with an arbitrary endofunctor H on Set (or, more generally, on every category with binary coproducts), which has free cias on every set Y. In Section 2, we shall introduce completely iterative algebras in this general setting. And we will prove the equivalence of the above statements (a) and (b). In Section 3, we prove an extension of the Solution Theorem of [1] to all completely iterative algebras. In Section 4, we prove (b) implies (c) (see above): Let H be an endofunctor on a category with binary coproducts (with monomorphic injections), which has free cias on every object Y. Then these free cias define a monad $\mathbb{T}$, and this monad is a free completely iterative monad on H. In Section 5, we show how the technical assumption of having monomorphic coproduct injections in the base category used in Section 4 can be avoided at the expense of being slightly more careful with some technical notions. This also leads to an extension of the freeness result. Finally, we shall prove in Section 6 that any free completely iterative monad is given by free completely iterative algebras, i.e., (c) implies (b) above. More precisely, if $\mathbb{T} = (T, \eta, \mu)$ is a free completely iterative monad on H, then for every object Y, TY is a free cia on Y, or, equivalently, TY is a final coalgebra of $H(\_) + Y$.

Related Work. The study of completely iterative algebras and completely iterative monads is very closely linked to the study of iterative algebras and iterative monads. In fact, historically, iterative theories were introduced by Elgot [9] before completely iterative theories. They are, roughly speaking, algebraic theories such that finitary recursive systems of equations, i.e., with a finite set of variables only, have unique solutions. Adámek et al. [2,3] have given a categorical approach to iterative theories. Similar ideas as those we use in the current paper for a simplified approach to completely iterative monads apply to the iterative case. In the latter case one starts by investigating iterative algebras, i.e., algebras that admit unique solutions of finitary systems of recursive equations. This leads to a construction of free iterative algebras using coalgebras, and these algebras yield the free iterative monad. This simplified approach to iterative theories can be found in [4]. That paper developed simultaneously with the current one.

In the classical setting of polynomial endofunctors on Set, iterative algebras were introduced by Nelson [18] to obtain a short proof of Elgot's description of free iterative theories. Also Tiuryn [20] introduced and studied a concept of iterative algebras with the aim of relating iterative theories to properties of algebras. Our notion of completely iterative algebras is an extension and generalization of the notion of iterative algebra of [18].

2. Completely iterative algebras for an endofunctor

Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be an endofunctor on a category $\mathcal{A}$ with binary coproducts. We denote by $\mathsf{inl} : X \longrightarrow X + Y$ and $\mathsf{inr} : Y \longrightarrow X + Y$ the coproduct injections and we shall write $\mathsf{can} : HX + HY \longrightarrow H(X + Y)$ for the canonical arrow $[H\mathsf{inl}, H\mathsf{inr}]$.

Definition 2.1. A morphism $e : X \longrightarrow HX + A$ of $\mathcal{A}$ is called a *flat equation morphism* in (the object of parameters) A. Suppose that A is the underlying object of an H-algebra $a : HA \longrightarrow A$. Then a *solution* of e in A is a morphism $e^\dagger : X \longrightarrow A$ such that the diagram

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ HX + A & \xrightarrow[He^\dagger + A]{} & HA + A \end{array} \tag{2.1}$$

commutes.

An H-algebra is called *completely iterative* (or shortly, *cia*) if every flat equation morphism in it has a unique solution.

Notation 2.2. For any flat equation morphism $e : X \longrightarrow HX + Y$ and any morphism $f : Y \longrightarrow Z$ we get a flat equation morphism $f \bullet e$ as the "renaming of parameters by f":

$$f \bullet e \equiv X \xrightarrow{e} HX + Y \xrightarrow{HX+f} HX + Z.$$

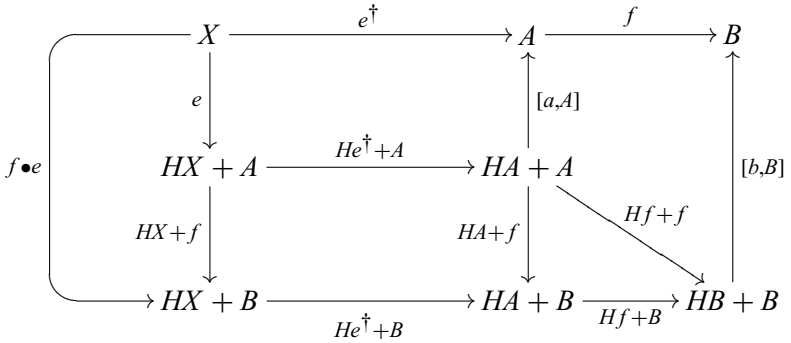
Homomorphisms of H-algebras are precisely the solution-preserving morphisms as we prove now:

Proposition 2.3. *Let (A, a) and (B, b) be completely iterative H-algebras, and let $f : A \longrightarrow B$ be a morphism. Then the following are equivalent:*

(i) $f : (A, a) \longrightarrow (B, b)$ is an H-algebra homomorphism,
(ii) f is solution-preserving, i.e., for all $e : X \longrightarrow HX + A$ we have

$$(f \bullet e)^\dagger = f \cdot e^\dagger.$$

Proof. (i) $\Rightarrow$ (ii): Consider the following commutative diagram:



In fact, the upper middle square commutes since $e^\dagger$ is a solution of e, and the upper right-hand part since f is an H-algebra homomorphism. The other three parts are obvious. Thus, the outer square commutes proving that $f \cdot e^\dagger$ is a solution of $f \bullet e$. The result follows from the unicity of solutions in B.

(ii) $\Rightarrow$ (i): Suppose that $f : A \longrightarrow B$ is a solution-preserving morphism. We have to show that f is an H-algebra homomorphism, i.e., $f \cdot a = b \cdot Hf$. To prove it we use the uniqueness of solutions. First, consider the equation morphism

$$e \equiv HA + A \xrightarrow{H\mathsf{inr}+A} H(HA + A) + A.$$

Its unique solution is $[a, A] : HA + A \longrightarrow A$. In fact, the following diagram

$$\begin{array}{ccc} HA + A & \xrightarrow{[a,A]} & A \\ {\scriptstyle H\mathsf{inr}+A}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ H(HA + A) + A & \xrightarrow[H[a,A]+A]{} & HA + A \end{array}$$

commutes. Since f is solution-preserving we know that $f \cdot a$ is the left-hand component of the unique solution of the following equation morphism:

$$f \bullet e \equiv HA + A \xrightarrow{H\mathsf{inr}+A} H(HA + A) + A \xrightarrow{H(HA+A)+f} H(HA + A) + B,$$

in symbols, $f \cdot a = (f \bullet e)^\dagger \cdot \mathsf{inl}$. Now consider the following commutative diagram:

$$\begin{array}{ccccc} HA + A & \xrightarrow{Hf+f} & HB + B & \xrightarrow{[b,B]} & B \\ {\scriptstyle H\mathsf{inr}+A}\downarrow & \searrow^{H(\mathsf{inr}\cdot f)+f} & & & \\ H(HA + A) + A & & \downarrow{\scriptstyle H\mathsf{inr}+B} & & \uparrow{\scriptstyle [b,B]} \\ {\scriptstyle H(HA+A)+f}\downarrow & & & \searrow & \\ H(HA + A) + B & \xrightarrow[H(Hf+f)+B]{} & H(HB + B) + B & \xrightarrow[H[b,B]+B]{} & HB + B \end{array}$$

It shows that $[b, B] \cdot (Hf + f) = (f \bullet e)^\dagger$; thus, we obtain

$$f \cdot a = (f \bullet e)^\dagger \cdot \mathsf{inl} = b \cdot Hf,$$

which completes the proof. □

Notation 2.4. We denote by $\mathsf{CIA}\,H$ the category of all completely iterative algebras and H-algebra homomorphisms. It is a full subcategory of $\mathsf{Alg}\,H$, the category of all H-algebras and homomorphisms.

Examples 2.5.

(i) Classical algebras are seldom cias. For example, let $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$ be the functor expressing one binary operation, $H_\Sigma X = X \times X$. Then a group is a cia iff its unique element is the unit 1, since the recursive equation $x \approx x \cdot 1$ has a unique solution. A lattice is a cia iff it has a unique element; consider $x \approx x \vee x$.

(ii) In [4] it was proved that the algebra of addition on

$$\widetilde{\mathrm{N}} = \{1, 2, 3, \ldots\} \cup \{\infty\}$$

is a cia w.r.t. the functor H_Σ of (i).

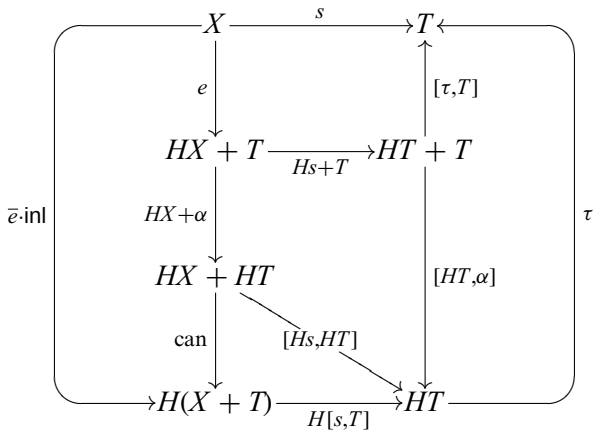
(iii) Final coalgebras are completely iterative algebras. More precisely, denote by (T, α) a final coalgebra of H, i.e., for any coalgebra (C, γ) there exists a unique coalgebra homomorphism $\gamma^\sharp : (C, \gamma) \longrightarrow (T, \alpha)$ so that $\alpha \cdot \gamma^\sharp = H(\gamma^\sharp) \cdot \gamma$. Recall that by Lambek's Lemma [12], the structure map α is an isomorphism, whose inverse we denote by $\tau : HT \longrightarrow T$. Then this H-algebra (T, τ) is completely iterative. In fact, consider an equation morphism

$$e : X \longrightarrow HX + T,$$

and form the H-coalgebra

$$\overline{e} \equiv X + T \xrightarrow{[e,\mathsf{inr}]} HX + T \xrightarrow{HX+\alpha} HX + HT \xrightarrow{\mathrm{can}} H(X + T).$$

We claim that the left-hand component of $\overline{e}^\sharp : X + T \longrightarrow T$ is the desired solution of e, and that it is unique. Indeed, any coalgebra homomorphism $(X + T, \overline{e}) \longrightarrow (T, \alpha)$ must have as its right-hand component a coalgebra homomorphism from (T, α) to itself, whence the identity on T. Then we get the following commutative diagram for the left-hand component:



If s is the left-hand component of $\overline{e}^\sharp$, then the outer shape commutes, whence so does the upper square, which shows that s solves e. Conversely, if s is a solution of e, then the upper square

commutes and therefore the outer shape does, too. Thus, $[s,T]: X+T \longrightarrow T$ is a coalgebra homomorphism, and so we have $[s,T] = \overline{e}^{\sharp}$.

(iv) Infinite trees form completely iterative algebras. Let Σ be a signature. It is well-known that the Σ-algebra T_Σ of all (finite and infinite) Σ-trees is a final H_Σ-coalgebra. Thus, T_Σ is a cia.

(v) Finitely branching strongly extensional trees. The final coalgebra of $\mathcal{P}_{\mathrm{fin}} : \mathsf{Set} \longrightarrow \mathsf{Set}$, the finite power-set functor, has been described by Worrell [21]. It is the algebra T of all strongly extensional finitely branching trees (i.e., unordered trees such that the subtrees defined by any pair of siblings are not bisimilar). It follows from (iii) that T is a cia.

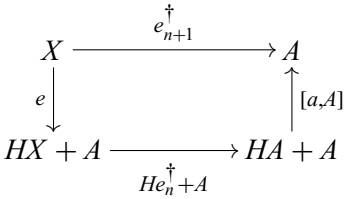
(vi) Algebras over complete metric spaces as a tool for the semantics of infinite computation have been investigated by America and Rutten [6]. Those algebras yield cias. Take $\mathcal{A} = \mathsf{CMS}$, the category whose objects are complete metric spaces (i.e., such that each Cauchy sequence has a limit), where distances are measured in the interval $[0,1]$. The morphisms of CMS are the non-expanding maps, i.e., functions $f : (X, d_X) \longrightarrow (Y, d_Y)$ such that $d_Y(f(x), f(y)) \leqslant d_X(x,y)$ for all $x, y \in X$. Recall that for given complete metric spaces (X, d_X) and (Y, d_Y) the hom-set in CMS is a complete metric space with the metric given by

$$d_{X,Y}(f,g) = \sup_{x \in X} d_Y(f(x), g(x)).$$

Now suppose we have a functor $H : \mathsf{CMS} \longrightarrow \mathsf{CMS}$ which is *contracting*, i.e., there exists a constant $\varepsilon < 1$ such that for any non-expanding maps $f, g : (X, d_X) \longrightarrow (Y, d_Y)$ between complete metric spaces we have

$$d_{HX,HY}(Hf, Hg) \leqslant \varepsilon \cdot d_{X,Y}(f,g).$$

Then any non-empty H-algebra (A,a) is completely iterative. In fact, given any flat equation morphism $e : X \longrightarrow HX + A$ in CMS, choose some element $a \in A$ and define a Cauchy sequence $(e^\dagger_n)_{n \in \mathbb{N}}$ in $\mathsf{CMS}(X,A)$ inductively as follows: let $e^\dagger_0 = \mathsf{const}_a$, and given $e^\dagger_n$ define $e^\dagger_{n+1}$ by the commutativity of the following diagram:



In [5] it is proved that this is indeed a Cauchy sequence in $\mathsf{CMS}(X,A)$ and that its limit yields a unique solution of e.

(vii) (Unary algebras over Set)
Here we have $\mathcal{A} = \mathsf{Set}$ and $H = Id$. A unary algebra (A, α_A) is completely iterative if and only if

(a) there exists a unique fixed point $a_0 \in A$ of all $\alpha_A^k : A \longrightarrow A, k \geqslant 1$,
(b) for any sequence $(b_i)_{i<\omega}$ in A with $b_i = \alpha_A(b_{i+1})$ we have $b_i = a_0$ for every $i < \omega$ (i.e., for any $a \neq a_0$ in A there is no infinite α-chain of elements of A ending in a).

To see that (a) and (b) are necessary, solve the equation $x \approx \alpha x$ to obtain the fixed point a_0. Furthermore, the system

$$x_i \approx \alpha x_{i+1}, \quad i < \omega,$$

has as solutions any sequence as in (b); in particular, the constant sequence at a_0 is a solution, and this must be the unique one.

For the sufficiency, suppose that (A, α_A) satisfies (a) and (b). Given any equation morphism $e : X \longrightarrow H_\Sigma X + A$ there is a unique solution $e^\dagger : X \longrightarrow A$: If $x \in X$ is such that there exist equations

$$\begin{aligned} x = x_0 &\approx \alpha x_1 \\ x_1 &\approx \alpha x_2 \\ &\vdots \\ x_{k-1} &\approx \alpha x_k \\ x_k &\approx a, \end{aligned}$$

where $a \in A$, then $e^\dagger(x_k) = a$ and therefore $e^\dagger(x) = \alpha^k(a)$. Otherwise we have equations

$$\begin{aligned} x = x_0 &\approx \alpha x_1 \\ x_1 &\approx \alpha x_2 \\ x_2 &\approx \alpha x_3 \\ &\vdots \end{aligned}$$

and (a) and (b) ensure that the unique solution is given by $e^\dagger(x_i) = a_0$, for all i.

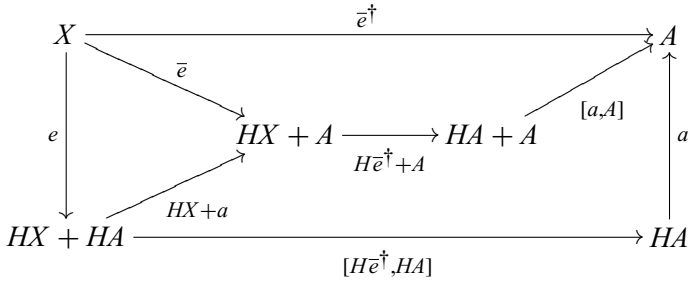
We shall now show that final H-coalgebras are precisely the initial completely iterative H-algebras. This is the first step towards proving the equivalence of the statements (a) and (b) of the introduction. First, we establish two auxiliary results. For the first one observe that any endofunctor H lifts to one on the category $\mathsf{Alg}\, H$ of algebras. The lifted endofunctor acts on objects by $(A, a) \longmapsto (HA, Ha)$, and on morphisms its action is that of H. The same is true for completely iterative algebras.

Proposition 2.6. *Any endofunctor H lifts to the category of completely iterative H-algebras, i.e., for any cia (A, a) the H-algebra (HA, Ha) is completely iterative, too.*

Proof. Suppose we are given an equation morphism $e : X \longrightarrow HX + HA$, we have to produce a solution $e^\dagger : X \longrightarrow HA$, and show its uniqueness. Let us form an equation morphism

$$\bar{e} \equiv X \xrightarrow{e} HX + HA \xrightarrow{HX + a} HX + A$$

w.r.t. (A, a). Then its solution $\overline{e}^\dagger$ makes the diagram

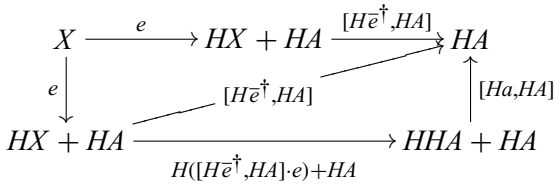


(2.2)

commutative. In fact, its upper part commutes since $\overline{e}^\dagger$ is a solution, and the other two parts are obvious. Now define

$$e^\dagger \equiv X \xrightarrow{e} HX + HA \xrightarrow{[H\overline{e}^\dagger, HA]} HA.$$

We prove that $e^\dagger$ solves e. In fact, the diagram



commutes; the upper left-hand triangle is obvious, and so is the right-hand coproduct component of the lower right-hand one. The left-hand coproduct component of the latter triangle yields the outer square of Diagram (2.2) after H is removed. This proves the existence of a solution.

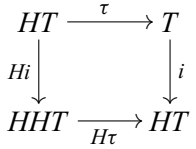
For the uniqueness, suppose that $s : X \longrightarrow HA$ solves e. Then $a \cdot s$ solves $\overline{e}$. In fact, notice that $a : (HA, Ha) \longrightarrow (A, a)$ is an H-algebra homomorphism and then use a similar argument as in the first part of the proof of Proposition 2.3. Thus, by uniqueness of solutions we have $a \cdot s = \overline{e}^\dagger$, and we obtain

$$\begin{aligned} s &= [Ha, HA] \cdot (Hs + HA) \cdot e \\ &= [H(a \cdot s), HA] \cdot e \\ &= [H\overline{e}^\dagger, HA] \cdot e \\ &= e^\dagger. \qquad \square \end{aligned}$$

Lambek's Lemma [12] states that the structure map of an initial H-algebra is an isomorphism. The same is true in the completely iterative case.

Lemma 2.7. *If (T, τ) is an initial completely iterative H-algebra, then the structure morphism τ is an isomorphism.*

Proof. By Proposition 2.6 we have a cia $(HT, H\tau)$. Then by initiality we obtain a unique H-algebra homomorphism $i : (T, \tau) \longrightarrow (HT, H\tau)$, i.e., such that the following square:



commutes. Clearly, $\tau : (HT, H\tau) \longrightarrow (T, \tau)$ is an H-algebra homomorphism. Thus, by initiality we conclude that $\tau \cdot i = 1_T$. But then also $i \cdot \tau = H\tau \cdot Hi = H1_T = 1_{HT}$. □

We are now ready to prove the main result of this section.

Theorem 2.8. *Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be any endofunctor.*

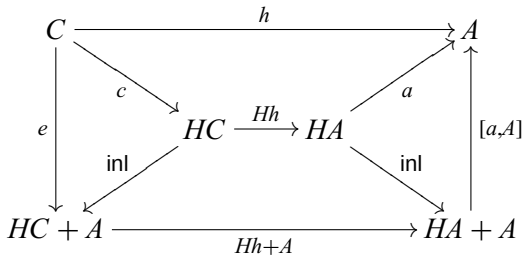
(i) If (T, α) is a final H-coalgebra, then (T, τ) with $\tau = \alpha^{-1}$ is an initial completely iterative H-algebra.

(ii) Conversely, if (T, τ) is an initial completely iterative H-algebra, then (T, α) with $\alpha = \tau^{-1}$ is a final H-coalgebra.

Proof. Before we prove the two statements we shall establish one useful fact about the relation between H-coalgebras and cia's. Suppose that (C, c) is any H-coalgebra and (A, a) is a cia. We can form an equation morphism

$$e \equiv C \xrightarrow{c} HC \xrightarrow{\mathsf{inl}} HC + A.$$

Then there is a one-to-one correspondence between solutions of e and morphisms $h : C \longrightarrow A$ such that $h = a \cdot Hh \cdot c$ (the so-called coalgebra to algebra homomorphisms). Indeed, this follows easily by inspection of the following diagram:



Since there exists a unique solution $e^\dagger$ for e, there exists a unique coalgebra to algebra homomorphism h. It is now quite easy to prove the theorem.

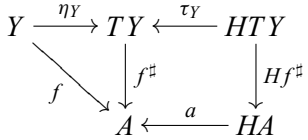
(i) We have seen in Example 2.5 that (T, τ) is completely iterative. It remains to prove the initiality. Given any cia (A, a) we have by the above considerations a unique coalgebra to algebra homomorphism $h : T \longrightarrow A$, i.e., unique H-algebra homomorphism $h : (T, \tau) \longrightarrow (A, a)$.

(ii) By Lemma 2.7 we only need to show finality of the coalgebra (T,α). Given any H-coalgebra (C,c) there exists a unique coalgebra to algebra homomorphism $h : C \longrightarrow T$, i.e., a unique H-coalgebra homomorphism $h : (C,c) \longrightarrow (T,\alpha)$. □

Remark 2.9. Observe that in the above proof of part (ii) in lieu of the full universal property of (T,τ) we have only used that the structure map τ is an isomorphism. Thus, the only cia with an isomorphic structure map is the initial one.

In the realm of H-algebras it is quite trivial to show that the initial algebra for the functor $H(\_) + Y$ is precisely the free H-algebra on the object Y. The same will now be proved for cia's, and this is the second neccessary ingredient to establish the equivalence of statements (a) and (b) from the introduction.

By a free cia on an object Y of $\mathcal{A}$ we mean, of course, a cia (TY,τ_Y) together with a morphism $\eta_Y : Y \longrightarrow TY$ in $\mathcal{A}$ such that for any cia (A,a) and any morphism $f : Y \longrightarrow A$ in $\mathcal{A}$ there exists a unique homomorphic extension $f^\sharp : (TY,\tau_Y) \longrightarrow (A,a)$, i.e., such that the diagram



commutes.

Theorem 2.10. *For any object Y of $\mathcal{A}$ the following are equivalent:*
(*i*) *TY is an initial completely iterative $H(\_) + Y$-algebra.*
(*ii*) *TY is a free completely iterative H-algebra on Y.*

Proof. First, we shall establish the following fact: To give a completely iterative H-algebra (A,a) and a morphism $f : Y \longrightarrow A$ is the same as to give a completely iterative algebra $(A,[a,f])$ of $H(\_) + Y$.

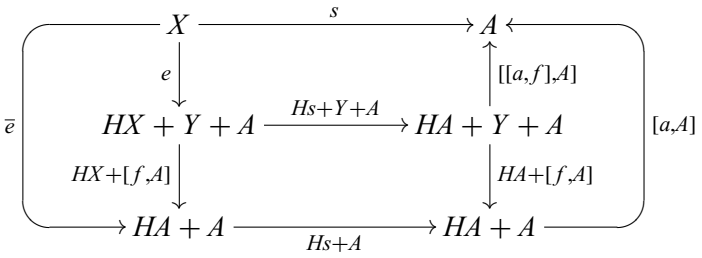
In fact, suppose we have a cia (A,a) and a morphism f. Then it is our task to find a unique solution for any equation morphism

$$e : X \longrightarrow HX + Y + A$$

for the functor $H(\_) + Y$. But e gives the following equation morphism

$$\overline{e} \equiv X \xrightarrow{\ e\ } HX + Y + A \xrightarrow{\ HX+[f,A]\ } HX + A$$

for the functor H. Now the solutions of e correspond precisely to the solutions of $\overline{e}$. Indeed, this follows by inspecting the following diagram:



The arrow s solves e if and only if the upper part commutes. Equivalently, the outer square commutes. But this says precisely that s solves $\overline{e}$. Since $\overline{e}$ has a unique solution, so has e.

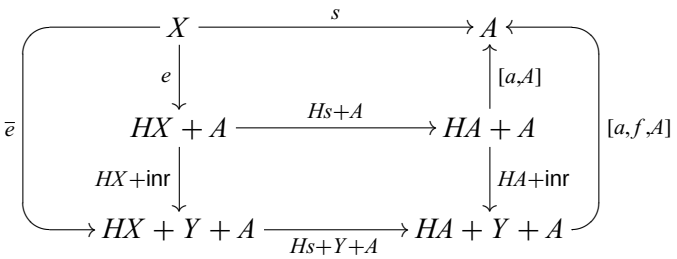
For the converse, suppose that $(A, [a, f])$ is a completely iterative algebra of $H(\_) + Y$. We must show that any equation morphism

$$e : X \longrightarrow HX + A$$

has a unique solution. We simply form an equation morphism

$$\overline{e} \equiv X \xrightarrow{e} HX + A \xrightarrow{HX+\mathsf{inr}} HX + Y + A.$$

As before, solutions of e correspond precisely to solutions of $\overline{e}$. In fact, inspect the following diagram:



The morphism s solves e precisely if the upper square commutes. This is equivalent to the commutativity of the outer shape, i.e., s solves $\overline{e}$. Hence, since $\overline{e}$ has a unique solution, so has e.

The result of the current theorem can now be proved precisely as in the case of ordinary H-algebras. This is straightforward and we leave it to the reader. □

In [l] we have called an endofunctor *iteratable*, if for any object Y of $\mathcal{A}$ there exists a final coalgebra TY of $H(\_) + Y$. Collecting the results of Theorems 2.8 and 2.10 we obtain the following characterization, i.e., the equivalence of statements (a) and (b), see Section 1.

Corollary 2.11. *For any endofuntor $H : \mathcal{A} \longrightarrow \mathcal{A}$ the following are equivalent:*

(*i*) *H is iteratable with final coalgebras TY of $H(\_) + Y$, for any Y in $\mathcal{A}$.*

(*ii*) *For any object Y there exists a free completely iterative H-algebra TY on Y.*

Example 2.12. *The free cias of* $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$.
Recall from Example 2.5(iv) the algebra T_Σ of all (finite and infinite) Σ-trees. This algebra is a cia. For every set Y the algebra $T_\Sigma Y$ of all Σ-trees over Y (i.e., trees with nodes having $n > 0$ children labelled by n-ary operation symbols and leaves labelled by constant symbols or variables from the set Y) is also a cia. It is well known that $T_\Sigma Y$ is a final coalgebra of $H_\Sigma(\_) + Y$. By Corollary 2.11, this implies that $T_\Sigma Y$ is a free completely iterative Σ-algebra on Y.

Example 2.13. *The free cias of* $\mathcal{P}_{\mathrm{fin}} : \mathsf{Set} \longrightarrow \mathsf{Set}$.
Recall the final coalgebra T of $\mathcal{P}_{\mathrm{fin}}$ from Example 2.5(v). Analogously, for a set Y a final coalgebra of $\mathcal{P}_{\mathrm{fin}}(\_) + Y$ is the algebra $T(Y)$ of all finitely branching strongly extensional trees with leaves partially labelled in the set Y. By Corollary 2.11, this implies that $T(Y)$ is a free cia on Y.

Remark 2.14. A special case of a recursive equation morphism is that where no parameters appear, i.e., simply coalgebras $e : X \longrightarrow HX$. They appear in various contexts, e.g., in non-wellfounded set theory [7] or, dually, in the theory of transitive sets [19]. However, these special equation morphisms are not sufficient for our purposes. Let us (just in the present remark) call an algebra *weakly iterative* if every equation morphism $e : X \longrightarrow HX$ has a unique solution $e^\dagger : X \longrightarrow A$ (i.e., $e^\dagger = a \cdot He^\dagger \cdot e$). For example in case $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$ represents a binary operation, $H_\Sigma X = X \times X$, the free cia $T_\Sigma\{a\}$ on one generator has the property that every equation $e : X \longrightarrow X \times X$ has the unique solution $e^\dagger : x \longmapsto t_0$, the constant function to the complete binary tree t_0. Consequently, every subalgebra of $T_\Sigma\{a\}$ containing t_0 is weakly iterative. However, not every such subalgebra is completely iterative; for example, the smallest subalgebra of $T_\Sigma\{a\}$ containing t_0 and all finite Σ-trees is weakly iterative but not completely iterative.

3. The solution theorem

In Section 1, we considered non-flat system (1.1) of formal recursive equations for Σ-algebras. And we argued that, due to the possibility of flattening such a system it suffices to consider only the flat equation morphisms $X \longrightarrow H_\Sigma X + A$. In this section, we shall make that statement precise by showing that in completely iterative algebras (not only in Set) much more general systems of recursive equations are uniquely solvable. This result illustrates that for polynomial endofunctors on Set cias are an extension and generalization of iterative algebras as presented by Nelson [18]. Applied to free cias our result implies the solution theorem of [1], which was also discovered independently by Moss [17] under the name Parametric Corecursion.

Let us remark first that the condition stated in (1.1) that no right-hand side of a system is a variable is important; for example, the equation $x \approx x$ has a unique solution only in the trivial terminal algebra. Systems satisfying the above condition are called *guarded*.

In this section we assume that $H : \mathcal{A} \longrightarrow \mathcal{A}$ is an iteratable endofunctor on a category $\mathcal{A}$ with binary coproducts. By Corollary 2.11, there exists a free cia TY on every object Y. In other words, we have an adjoint situation

$$\mathsf{CIA}\, H \underset{\longrightarrow}{\overset{\longleftarrow}{\perp}} \mathcal{A}.$$

This adjunction creates a monad $\mathbb{T} = (T, \eta, \mu)$ on $\mathcal{A}$. More detailed, for every object Y denote by TY the (underlying object of a) free cia on Y with universal arrow

$$\eta_Y : Y \longrightarrow TY$$

and algebra structure

$$\tau_Y : HTY \longrightarrow TY.$$

Use the freeness of the cia TTY on TY to obtain $\mu_Y : TTY \longrightarrow TY$ as the unique homomorphism of H-algebras with $\mu_Y \cdot \eta_{TY} = 1_{TY}$. It is easy to check the naturality of η, τ, and μ as well as the three monad laws. Notice also that it follows from Theorems 2.8 and 2.10 that the morphism $[\tau_Y, \eta_Y] : HTY + Y \longrightarrow TY$ is an isomorphism whose inverse is the structure map of a final coalgebra of $H(\_) + Y$.

Finally, observe that the following Substitution Theorem proved in [1] using coinduction is now a trivial consequence of the freeness of the cias TY:

Theorem 3.1 (Substitution theorem)**.** *For any morphism $s : X \longrightarrow TY$ there exists a unique homomorphism $\widehat{s} : TX \longrightarrow TY$ of H-algebras extending s, i.e., with $\widehat{s} \cdot \eta_X = s$.*

Remark 3.2. In case of a polynomial endofunctor on Set induced by a signature Σ Theorem 3.1 states that substitution works for infinite Σ-trees in precisely the same way as for terms (i.e., finite trees): for a set X of variables the mapping $s : X \longrightarrow T_\Sigma Y$ assigns to each variable its substitute, which is a Σ-tree over the set Y, and the extension $\widehat{s} : T_\Sigma X \longrightarrow T_\Sigma Y$ performs on any tree t of $T_\Sigma X$ the substitution s, to obtain a tree of $T_\Sigma Y$.

Notice that the fact that each $\widehat{s}$ is an H-algebra homomorphism results in the following property of substitution of infinite trees: for each tree t which is not just a leaf labelled by a variable, i.e., for all elements of the left-hand coproduct component $H_\Sigma T_\Sigma X$ of $T_\Sigma X$, the result of any substitution will never be just a leaf labelled in Y, i.e., $\widehat{s}(t)$ lies in $H_\Sigma T_\Sigma Y$. Or, more shortly, non-variables are preserved by substitution.

Whereas the concept of variables and substitution is appropriately captured categorically by the concept of a monad, the idea of "non-variable" and its preservation by substitution is not. However, we will need such a concept when we speak of guarded systems of equations below. In fact, in the setting of algebraic theories (i.e., monads on Set) Elgot [9] introduced the concept of an ideal theory. In [1] we proved that the following concept is equivalent to this.

For a monad $\mathbb{S} = (S, \eta, \mu)$ over Set we can form the complements of the image $\eta_X[X]$ of X under η_X in SX, say,

$$\sigma_X : S'X \longrightarrow SX$$

for all objects X.

The monad is called *ideal* provided $\sigma : S' \longrightarrow S$ is a subfunctor of S, and the monad multiplication has a domain-codomain restriction $\mu' : S'S \longrightarrow S'$. For general base categories the corresponding concept is as follows:

Definition 3.3. By an *ideal monad* is understood a six-tuple

$$\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$$

consisting of a monad (S, η, μ), a subfunctor $\sigma : S' \hookrightarrow S$ and a natural transformation $\mu' : S'S \longrightarrow S'$ such that

(i) $S = S' + Id$ with coproduct injections σ and η, and
(ii) μ restricts to μ' along σ, i.e., the following square

$$\begin{array}{ccc} S'S & \xrightarrow{\mu'} & S' \\ {\scriptstyle \sigma S}\downarrow & & \downarrow{\scriptstyle \sigma} \\ SS & \xrightarrow[\mu]{} & S \end{array}$$

commutes.

Examples 3.4.

(i) Free monads are ideal. If H is a *varietor*, i.e., there exist free H-algebras FY on every object Y of $\mathcal{A}$, then this is the object assignment of a free monad F on H, and this monad is ideal. In fact, it is well-known that we have a coproduct $FY = HFY + Y$ with injections $\varphi_Y : HFY \longrightarrow FY$ and $\eta_Y : Y \longrightarrow FY$ given by the structure and the universal arrow of the free H-algebra. Thus, since coproduct injections are monomorphic, we have the subfunctor

$$\varphi : HF \hookrightarrow F.$$

The restriction of μ is

$$\mu' = H\mu : HFF \longrightarrow HF$$

and the square

$$\begin{array}{ccc} HFF & \xrightarrow{H\mu} & HF \\ {\scriptstyle \varphi F}\downarrow & & \downarrow{\scriptstyle \varphi} \\ FF & \xrightarrow[\mu]{} & F \end{array}$$

commutes since μ_Y is defined as the unique H-algebra homomorphism with $\mu_Y \cdot \eta_{FY} = 1_{FY}$.

(ii) Similarly, the free cia monad $\mathbb{T} = (T, \eta, \mu)$ together with the endofunctor HT and the natural transformation

$$\tau : HT \hookrightarrow T$$

expressing the H-algebra structure $\tau_Y : HTY \longrightarrow TY$ of each TY is ideal. The restriction of μ is $\mu' = H\mu$ again.

(iii) The monad on Set given by the free algebras with a binary commutative operation is ideal. In fact, this is the free monad on the endofunctor that assigns to every set X the set of unordered pairs from X.
(iv) The free semigroup monad $X \longmapsto X^+$ on Set is ideal. Here $S'X \hookrightarrow X^+$ is the subset of words of length at least 2, and μ' is the obvious restriction of the concatenation of words to that subset.
(v) The free monoid monad $X \longmapsto X^*$ on Set is not ideal. In fact, recall that the unit η_X maps elements of X to words of length 1. Now consider the word xx' in $\{x, x'\}^*$ and the substitution s that subsitutes x by itself and x' by the empty word. Then $\widehat{s}(xx') = x$ whence μ cannot have the necessary restriction.
(vi) Classical algebraic theories (groups, lattices, etc.) are usually not ideal.
(vii) For a polynomial endofunctor on Set the algebras $R_\Sigma Y$ of *rational trees*, i.e., those finite and infinite Σ-trees over Y that have (up to isomorphism) finitely many subtrees only, yield an ideal monad R_Σ on Set. More generally, we have shown in [4] that any finitary functor H on a locally presentable category $\mathcal{A}$ generates a rational monad R, and that this monad is ideal.
(viii) Coproducts of ideal monads exist and are ideal. Assume that $\mathcal{A}$ has colimits of ω-chains and let $S = S' + Id$ and $M = M' + Id$ be ideal monads so that S' and M' are ω-cocontinuous, i.e., they preserve colimits of ω-chains. Then a coproduct of S and M in the category of monads of $\mathcal{A}$ exists and is an ideal monad, see [11].

Remark 3.5. In [1] we defined an *equation morphism* to be a morphism

$$e : X \longrightarrow T(X + Y)$$

generalizing and extending the notion of a non-flat system, see (1.1). An equation morphism e is called *guarded* whenever there exists a factorization through $[\tau_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}]$:

$$\begin{array}{ccc} X & \xrightarrow{\ e\ } & T(X+Y) \\ & \searrow & \uparrow{\scriptstyle [\tau,\eta\cdot \mathsf{inr}]} \\ & & HT(X+Y)+Y \end{array}$$

We proved that any guarded equation morphism has a unique solution, i.e., a unique morphism $e^\dagger : X \longrightarrow TY$ such that the square

$$\begin{array}{ccc} X & \xrightarrow{\ e^\dagger\ } & TY \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu_Y} \\ T(X+Y) & \xrightarrow[T[e^\dagger,\eta_Y]]{} & TTY \end{array} \tag{3.1}$$

commutes. It is easy to extend the notion of equation morphisms and their solution to any monad, and the notion of guardedness to any ideal monad $\mathbb{S}$, see [1], Definition 4.7. In the current paper, we go one step further, and we introduce solutions in any Eilenberg–Moore algebra of $\mathbb{S}$, and we prove that any cia considered as an algebra of $\mathbb{T}$ admits unique solutions of guarded equation morphisms.

Definition 3.6. Let $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ be an ideal monad on $\mathcal{A}$.

(i) By an *equation morphism* is meant a morphism

$$e : X \longrightarrow S(X + Y)$$

in $\mathcal{A}$ where X is any object ("of variables") and Y is any object ("of parameters").

(ii) The equation morphism e is called *guarded* if it factors through the morphism $[\sigma_{X+Y}, \eta_{X+Y} \cdot \mathsf{inr}]$:

$$\begin{array}{ccc} X & \xrightarrow{\ e\ } & S(X+Y) \\ & \searrow & \uparrow{\scriptstyle [\sigma,\eta\cdot\mathsf{inr}]} \\ & & S'(X+Y)+Y \end{array}$$

(iii) Given an Eilenberg–Moore algebra $\alpha : SA \longrightarrow A$ and a morphism $f : Y \longrightarrow A$ (interpreting parameters in A), we call a morphism $e^\dagger : X \longrightarrow A$ a *solution of e induced by f* provided that the square

$$\begin{array}{ccc} X & \xrightarrow{\ e^\dagger\ } & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \alpha} \\ S(X+Y) & \xrightarrow[S[e^\dagger, f]]{} & SA \end{array} \tag{3.2}$$

commutes.

Notation 3.7. For any cia $a : HA \longrightarrow A$ we denote by

$$\widetilde{a} : TA \longrightarrow A$$

the unique H-algebra homomorphism with $\widetilde{a} \cdot \eta_A = 1_A$. It is easy to check that $\widetilde{a}$ is the structure of an Eilenberg–Moore algebra of the monad $\mathbb{T}$. Notice that in case of a polynomial functor this can be thought of as computations of finite and infinite Σ-trees over A in the Σ-algebra A.

Remark 3.8. For the free cia monad $\mathbb{T}$ obtained from a polynomial functor of Set and a cia (A, a) considered as an Eilenberg–Moore algebra $\widetilde{a} : TA \longrightarrow A$ the commutativity of square (3.2) means that the assignment $e^\dagger$ of variables of X to elements of A has the following property: form first the "substitution" mapping $[e^\dagger, f] : X + Y \longrightarrow A$ (which interprets variables according to the solution $e^\dagger$ and parameters according to f). Apply this substitution to the right-hand side of the given system e of formal equations, and compute the resulting infinite trees in A. This yields the same assignment of variables to elements of A as $e^\dagger$. That means that the formal equations $x \approx e(x)$ become actual identities in A after the substitution $x \longmapsto e^\dagger(x)$ is performed on both sides of the equations and the right-hand side is evaluated in A.

Formally, one extends $[e^\dagger, f]$ to the unique homomorphism

$$\widetilde{a} \cdot T[e^\dagger, f] : T(X + Y) \longrightarrow A$$

from the free cia on $X + Y$ to A. Precomposed with e it yields the morphism $e^\dagger$.

Theorem 3.9. *In a completely iterative algebra, for any guarded equation morphism and every interpretation of its parameters there exists a unique solution.*

Remark. More precisely, let $a : HA \longrightarrow A$ be a cia considered as an Eilenberg–Moore algebra $\widetilde{a} : TA \longrightarrow A$. Suppose that we have a guarded equation morphism

$$\begin{array}{ccc} X & \xrightarrow{\ e\ } & T(X+Y) \\ & \searrow^{e_0} & \uparrow{\scriptstyle [\tau,\eta\cdot\mathsf{inr}]} \\ & & HT(X+Y)+Y \end{array} \tag{3.3}$$

and an interpretation $f : Y \longrightarrow A$. Then there exists a unique morphism $e^\dagger : X \longrightarrow A$ such that the square

$$\begin{array}{ccc} X & \xrightarrow{\ e^\dagger\ } & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \widetilde{a}} \\ T(X+Y) & \xrightarrow[T[e^\dagger,f]]{} & TA \end{array}$$

commutes.

Proof. We form the following flat equation morphism

$$\overline{e} \equiv T(X+Y) \xrightarrow{[\tau,\eta]^{-1}} HT(X+Y)+X+Y \xrightarrow{[\mathsf{inl},e_0,\mathsf{inr}]} HT(X+Y)+Y \xrightarrow{HT(X+Y)+f} HT(X+Y)+A$$

w.r.t. the cia A. Let us denote by s the unique solution of $\overline{e}$, i.e., s is the unique morphism such that the following diagram

$$\begin{array}{ccc} T(X+Y) & \xrightarrow{\ s\ } & A \\ {\scriptstyle [\tau,\eta]^{-1}}\downarrow\uparrow{\scriptstyle [\tau,\eta]} & & \\ HT(X+Y)+X+Y & & \\ {\scriptstyle [\mathsf{inl},e_0,\mathsf{inr}]}\downarrow & & \uparrow{\scriptstyle [a,A]} \\ HT(X+Y)+Y & & \\ {\scriptstyle HT(X+Y)+f}\downarrow & & \\ HT(X+Y)+A & \xrightarrow[Hs+A]{} & HA+A \end{array} \tag{3.4}$$

commutes. Consider the coproduct components of $HT(X+Y)+X+Y$ separately to conclude that s is uniquely determined by the following three equations:

(i) $s \cdot \tau_{X+Y} = a \cdot Hs$,
(ii) $s \cdot \eta_{X+Y} \cdot \mathsf{inr} = f$,
(iii) $s \cdot \eta_{X+Y} \cdot \mathsf{inl} = [a, A] \cdot (Hs + A) \cdot (HT(X + Y) + f) \cdot e_0$.

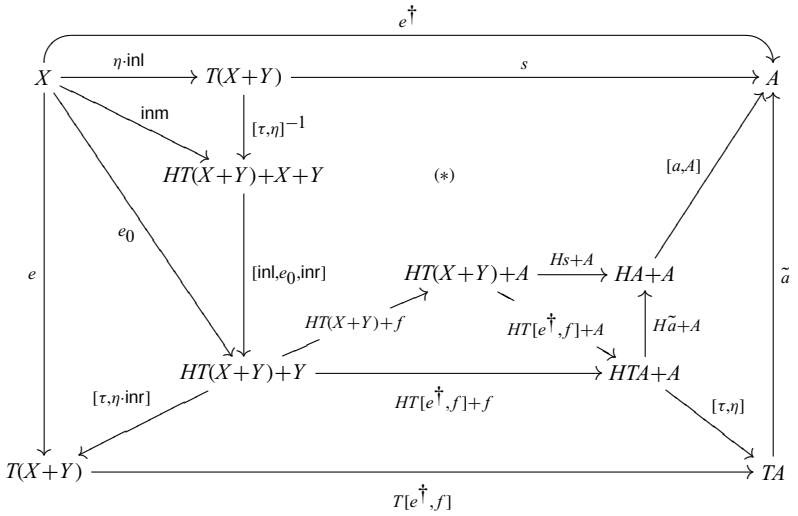
Existence of a solution of e induced by f. We define the morphism

$$e^\dagger \equiv X \xrightarrow{\mathsf{inl}} X + Y \xrightarrow{\eta_{X+Y}} T(X + Y) \xrightarrow{s} A.$$

It follows immediately that s is a homomorphism of H-algebras with $s \cdot \eta_{X+Y} = [e^\dagger, f] : X + Y \longrightarrow A$, see (i) and (ii) above. We invoke the freeness of the cia $T(X + Y)$ to conclude that the equation

$$s = \widetilde{a} \cdot T[e^\dagger, f] \tag{3.5}$$

holds. To establish that $e^\dagger$ is a solution of e induced by f consider the following commutative diagram



(3.6)

The upper most part is the definition of $e^\dagger$, part $(*)$ is diagram (3.4), the lower left-hand triangle is (3.3), and the right-hand part commutes due to the definition of $\widetilde{a}$. The inner right-hand triangle commutes because of (3.5), and the remaining parts are obvious. Thus, the outer square commutes as desired.

Uniqueness. Suppose that $e^\dagger$ is any solution of e induced by f. Let $s = \widetilde{a} \cdot T[e^\dagger, f] : T(X + Y) \longrightarrow A$. To complete the proof it suffices to show that s solves $\overline{e}$ w.r.t. the cia A (equivalently, the above equations (i), (ii), and (iii) hold). In fact, the morphism s determines $e^\dagger$ since clearly we have the equation $e^\dagger = s \cdot \eta_{X+Y} \cdot \mathsf{inl}$. Observe first that s is an H-algebra homomorphism with $s \cdot \eta_{X+Y} = [e^\dagger, f]$. Thus, the equations (i) and (ii) hold. To see that equation (iii) holds consider again diagram (3.6). Its outer square commutes since $e^\dagger$ is a solution of e induced by f. Since all other parts clearly commute, so does part $(*)$ when extended by $\eta_{X+Y} \cdot \mathsf{inl}$. But this is precisely the desired equation (iii). □

Remark 3.10. Theorem 3.9 shows that the Eilenberg–Moore algebra $\widetilde{a} : TA \longrightarrow A$ arising from a cia (A, a) admits unique solutions. For general Eilenberg–Moore algebras of $\mathbb{T}$ uniqueness of so-

lutions fails. In fact, for the endofunctor $H_\Sigma = Id$ of Set (expressing one unary operation) the two element set $\{0,1\}$ carries an Eilenberg–Moore algebra as follows: Notice that for any set X, $T_\Sigma X = \mathrm{N} \times X + \{\infty\}$. It is easy to check that the map

$$T_\Sigma A \longrightarrow A, \quad (n,i) \longmapsto i,\ i = 0,1, \quad \infty \longmapsto 0$$

is a structure of an Eilenberg–Moore algebra, see also [5], Example 3.8 and Theorem 5.5. However, the equation $x \approx x$ expressed by the guarded equation morphism

$$\{x\} \longrightarrow \mathrm{N} \times \{x\} + \{\infty\} = T_\Sigma(\{x\} + \emptyset), \quad x \longmapsto (1,x),$$

has for the unique interpretation $\emptyset \longrightarrow A$ two solutions 0 and 1.

The following result was first proved independently by Moss [17] and by Aczel et al. [1]. We obtain it as a corollary of Theorem 3.9.

Theorem 3.11 (Solution theorem)**.** *For any guarded equation morphism $e : X \longrightarrow T(X + Y)$ there exists a unique solution in the algebra TY, i.e., a unique morphism $e^\dagger : X \longrightarrow TY$ such that Diagram (3.1) commutes.*

Proof. Apply Theorem 3.9 to e and $f = \eta_Y : Y \longrightarrow TY$ and observe that $\widetilde{\tau_Y} = \mu_Y : TTY \longrightarrow TY$. □

4. Free completely iterative monad

In this section, we still assume that $H : \mathcal{A} \longrightarrow \mathcal{A}$ is an iteratable endofunctor on a category $\mathcal{A}$ with binary coproducts (equivalently, H has free cias on every object of $\mathcal{A}$, see Corollary 2.11). We also assume that coproduct injections are monomorphic; this can be avoided (see Section 5).

As the main result of [1] it was proved that the monad $\mathbb{T}$, which is given by the final coalgebras TY of $H(\_) + Y$, is a free completely iterative monad on H. The proof given there is technically quite complicated, involving an unpleasant amount of rather unintuitive diagram chasing arguments. Here we will give a much simpler proof. Recall the statements (a), (b), and (c) from the introduction. In lieu of proving (a) implies (c) directly we use the equivalence of (a) and (b) established in Section 2 and prove (b) implies (c). In fact, the universal property of the free cias TY, for every object Y, more easily yields the desired universal property of the monad $\mathbb{T}$.

We start by recalling the definition of a completely iterative monad from [1].

Definition 4.1. An ideal monad $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ is called *completely iterative* if every guarded equation morphism $e : X \longrightarrow S(X + Y)$ has a unique solution induced by $\eta_Y : Y \longrightarrow SY$ in the free Eilenberg–Moore algebra $\mu_Y : SSY \longrightarrow SY$, i.e., for each guarded e there exists a unique solution $e^\dagger : X \longrightarrow SY$ so that the square

$$\begin{array}{ccc} X & \xrightarrow{e^\dagger} & SY \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle \mu_Y} \\ S(X+Y) & \xrightarrow[S[e^\dagger,\eta_Y]]{} & SSY \end{array} \tag{4.1}$$

commutes.

An *ideal monad morphism* from an ideal monad $(S, \eta^S, \mu^S, S', \sigma, \mu'^S)$ to an ideal monad $(U, \eta^U, \mu^U, U', \omega, \mu'^U)$ is a monad morphism $\lambda : (S, \eta^S, \mu^S) \longrightarrow (U, \eta^U, \mu^U)$ which has a domain-codomain restriction to the ideals (i.e., there exists a natural transformation $\lambda' : S' \longrightarrow U'$ with $\lambda \cdot \sigma = \omega \cdot \lambda'$).

Given a functor H, a natural transformation $\lambda : H \longrightarrow S$ is called *ideal* provided that it factors through $\sigma : S' \hookrightarrow S$.

Example 4.2. The monad $\mathbb{T}$ is ideal (w.r.t. $T \cong HT + Id$), and the Solution Theorem 3.11 states that $\mathbb{T}$ is completely iterative. Notice that $\mathbb{T}$ comes with the following canonical natural transformation

$$\kappa \equiv H \xrightarrow{H\eta} HT \xrightarrow{\tau} T,$$

which is ideal.

Theorem 4.3. *The monad $\mathbb{T}$ is a free completely iterative monad. That is, for any completely iterative monad $\mathbb{S}$ and every ideal natural transformation $\lambda : H \longrightarrow S$ there exists a unique monad morphism $\overline{\lambda} : \mathbb{T} \longrightarrow \mathbb{S}$ with $\overline{\lambda} \cdot \kappa = \lambda$. And the induced $\overline{\lambda}$ is an ideal monad morphism.*

Remark 4.4.

(i) Notice that the statement of the Theorem is slightly stronger than in [1]. Here we do not require that the monad morphism $\overline{\lambda}$ be ideal in order to obtain its uniqueness. And the proof is substantially simpler.

(ii) For the category $\mathsf{CIM}(\mathcal{A})$ of all completely iterative monads and ideal monad morphisms we have a forgetful functor

$$U : \mathsf{CIM}(\mathcal{A}) \longrightarrow [\mathcal{A}, \mathcal{A}], \quad \mathbb{S} \longmapsto S'.$$

The theorem states that there exists a universal arrow at each iteratable endofunctor H. However, notice that this does not imply the existence of a left adjoint to U. (A left adjoint may not even exist if one restricts the codomain of U to iteratable functors. It is not clear that for an iteratable functor H the ideal HT of the completely iterative monad $\mathbb{T}$ is iteratable again.) If $\mathcal{A}$ is a locally presentable category and we restrict the codomain of U to $\mathsf{Acc}[\mathcal{A}, \mathcal{A}]$, the category of accessible endofunctors on $\mathcal{A}$, and the domain to the category $\mathsf{CIAM}(\mathcal{A})$ of accessible completely iterative monads $\mathbb{S}$ (i.e., such that both S and S' are accessible) then this restriction has a left adjoint, viz. the functor $H \longmapsto \mathbb{T}$. In fact, T is then accessible, see [4],

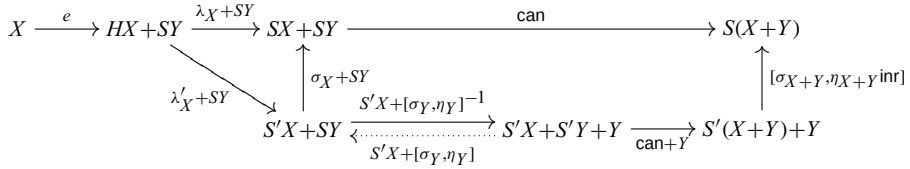
Proof. (1) For every object Y consider SY as an H-algebra as follows:

$$HSY \xrightarrow{\lambda_{SY}} SSY \xrightarrow{\mu_Y} SY.$$

It is completely iterative. In fact, every equation morphism $e : X \longrightarrow HX + SY$ yields the following equation morphism w.r.t. $\mathbb{S}$:

$$\overline{e} \equiv X \xrightarrow{e} HX + SY \xrightarrow{\lambda_X + SY} SX + SY \xrightarrow{\mathsf{can}} S(X + Y).$$

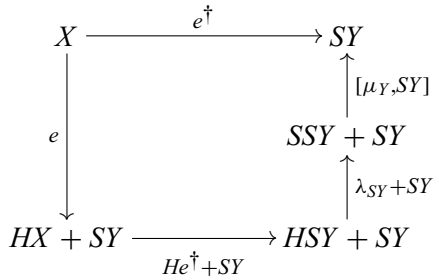
To verify that $\overline{e}$ is guarded, use the restriction $\lambda' : H \longrightarrow S'$ of λ and consider the commutative diagram



To see the commutativity of the square, consider the three components of $S'X + S'Y + Y$ separately, and use naturality of σ and η.

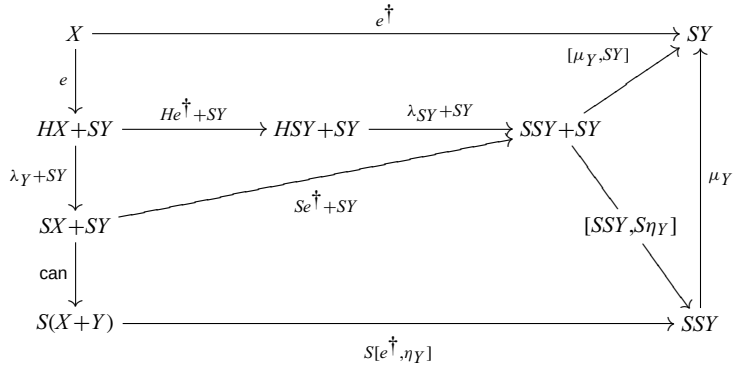
We prove that a morphism $e^\dagger : X \longrightarrow SY$ is a solution of e in the H-algebra SY if and only if it is a solution of $\overline{e}$ w.r.t. the iterative monad $\mathbb{S}$.

(1a) Let $e^\dagger$ be a solution of e in the algebra SY, i.e., let



(4.2)

commute. We are to show that the following diagram:



(4.3)

has the outward square commutative. The upper part is (4.2), the one directly below it is the naturality of λ. The lower part is obvious as is the right-hand triangle due to $\mu_Y \cdot S\eta_Y = 1_{SY}$.

(1b) Let the outward square of (4.3) commute. Then (4.2) commutes because it forms the upper part of (4.3), where the two adjacent parts and the lower part commute.

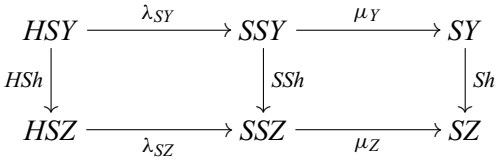
(2) *Existence of an ideal monad morphism* $\overline{\lambda}$ *with* $\overline{\lambda} \cdot \kappa = \lambda$. Denote by

$$\overline{\lambda}_Y : TY \longrightarrow SY$$

the unique homomorphism of H-algebras with

$\overline{\lambda}_Y \cdot \eta_Y = \eta_Y^S$.

We first observe that $\overline{\lambda}$ is a natural transformation. Given a morphism $h : Y \longrightarrow Z$ then Sh is a homomorphism of H-algebras from SY to SZ:

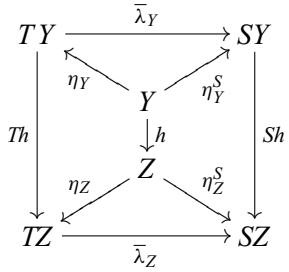


(4.4)

Thus, we have two parallel homomorphisms of H-algebras

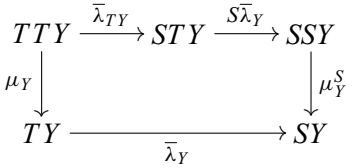
$Sh \cdot \overline{\lambda}_Y, \overline{\lambda}_Z \cdot Th : TY \longrightarrow SZ$.

They agree when precomposed with η_Y; in fact, the following diagram commutes:



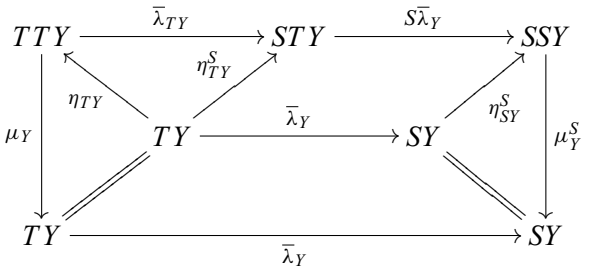
By the universal property of η_Y, and since SZ is a completely iterative H-algebra, this proves that the above naturality square commutes.

Let us prove that $\overline{\lambda}$ is a monad morphism. Since $\overline{\lambda} \cdot \eta = \eta^S$ by definition, it only remains to prove the commutativity of the following diagram:



(4.5)

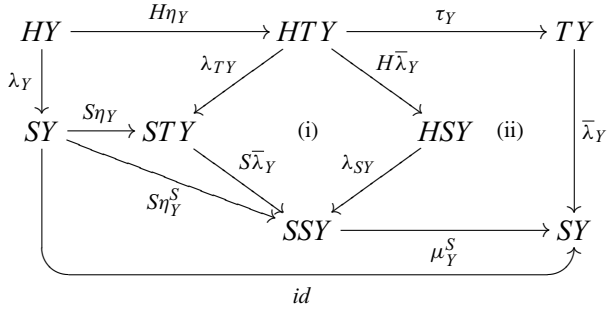
By (4.4), applied to $h = \overline{\lambda}_Y$, we see that $S\overline{\lambda}_Y$ is a homomorphism of H-algebras. By the universal property of η_{TY} it is sufficient to prove that (4.5) commutes when precomposed with η_{TY}:



Finally, the equation

$$\lambda = \overline{\lambda} \cdot \kappa = \overline{\lambda} \cdot \tau \cdot H\eta$$

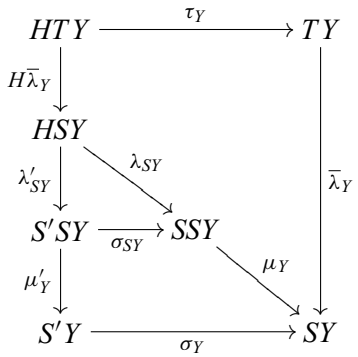
follows from the commutativity of the diagram



(4.6)

where (i) is naturality of λ, (ii) is the definition of $\overline{\lambda}$, the upper left-hand part is the naturality of λ, and the triangle below it uses the unit law for $\overline{\lambda}$. For the lowest part use the monad law $\mu^S_Y \cdot S\eta^S_Y = 1_{SY}$.

Thus, we have found a monad morphism $\overline{\lambda} : \mathbb{T} \longrightarrow \mathbb{S}$ with $\overline{\lambda} \cdot \kappa = \lambda$. It remains to verify that $\overline{\lambda}$ is ideal. To this end consider the commutative diagram



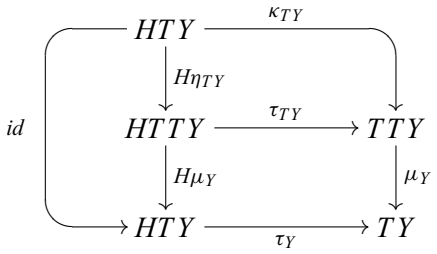
The upper right-hand part commutes by the definition of $\overline{\lambda}_Y$, the left-hand triangle commutes since λ is an ideal transformation, and for the lower part we use that μ restricts to μ'. Thus, we see that $\mu'^S \cdot \lambda' S \cdot H\overline{\lambda} : HT \longrightarrow S'$ is the desired restriction of $\overline{\lambda}$.

(3) *Uniqueness of* $\overline{\lambda}$. Suppose that $\overline{\lambda} : \mathbb{T} \longrightarrow \mathbb{S}$ is a monad morphism with $\overline{\lambda} \cdot \kappa = \lambda$. We are going to show that for any object Y, $\overline{\lambda}_Y$ is an H-algebra homomorphism extending η^S_Y, and then invoke the freeness of TY as a completely iterative H-algebra, which establishes the desired uniqueness.

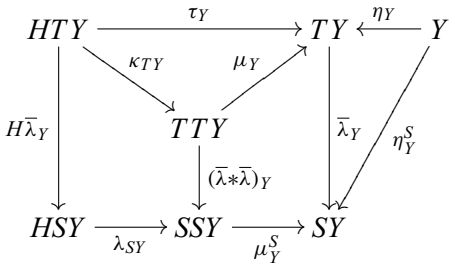
First, notice that for any object Y we have

$$\tau_Y = \mu_Y \cdot \kappa_{TY}. \tag{4.7}$$

Indeed, the following diagram commutes:



Consequently, the following diagram

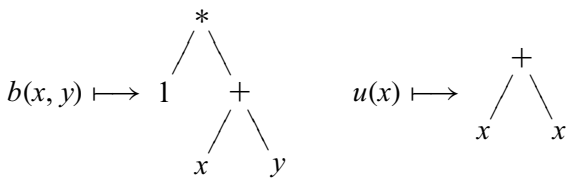


commutes: the right-hand triangle and the lower right-hand part commute since $\overline{\lambda}$ is a monad morphism, the lower left-hand part commutes since $\overline{\lambda} \cdot \kappa = \lambda$ and by naturality, and the upper triangle is (4.7).

Thus, $\overline{\lambda}_Y : TY \longrightarrow SY$ is an H-algebra homomorphism between completely iterative H-algebras such that $\overline{\lambda}_Y \cdot \eta_Y = \eta^S_Y$. This determines $\overline{\lambda}_Y$ uniquely. □

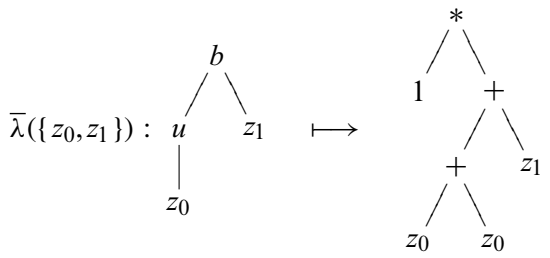
Remark 4.5. For polynomial endofunctors on Set, the freeness of $\mathbb{T}$ specializes to *second order substitution*, see [8], i.e., substitution of finite or infinite trees for operation symbols.

For example, consider a signature Σ with a binary operation symbol b, and a unary one u, and another signature Γ with two binary operation symbols $+$ and $*$ and a constant symbol 1. The following assignment:



(4.8)

of operation symbols in Σ to Γ-trees gives rise to a natural transformation $\lambda : H_\Sigma \longrightarrow T_\Gamma$. The induced ideal monad morphism $\overline{\lambda} : \mathbb{T}_\Sigma \longrightarrow \mathbb{T}_\Gamma$ replaces, for any set of variables X, the operation symbols in trees of $T_\Sigma X$ according to λ. Example:



The requirement that λ be an ideal transformation means that no operation symbol of Σ is replaced by a single variable, i.e., that λ is a so called *non-erasing* substitution.

5. Idealized monads

In this section, we show how to prove the results of the previous section in full generality, i.e., for any category $\mathcal{A}$ with binary coproducts (not necessarily having monomorphic coproduct injections) and every iteratable endofunctor $H : \mathcal{A} \longrightarrow \mathcal{A}$. The proof ideas remain essentially unchanged, although the technical difficulty is somewhat increased due to the fact that ideal monads should be replaced with *idealized monads*, which we introduce below, but, on the other hand, the freeness result of Theorem 4.3 can be extended a little further.

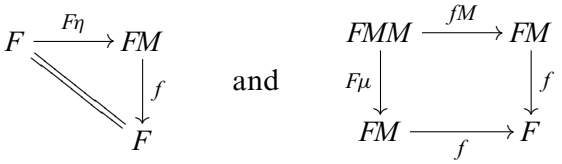
The main technical tool of this section is an ideal coreflection of any idealized monad. The ideas for the proofs of the respective results are essentially those used in the technical material of [16]. We shall need that material in Section 6 below where we complete the proof of the equivalence of the three statements (a), (b), and (c) from the introduction. Here we will use an ideal coreflection to extend the result of Theorem 4.3 to idealized monads, more precisely, we prove in Theorem 5.14 below that the free cia monad $\mathbb{T}$ is a free w.r.t. all idealized completely iterative monads, thus we establish (b) implies (c) in full generality.

Remark 5.1.

(i) Recall that in Example 3.4(ii) we showed that the free cia monad $\mathbb{T}$ is ideal. A quick inspection of all previous proofs reveals that this was the only place where we used the assumption that coproduct injections are monomorphic. However, when we drop that assumption, it is no longer sufficient to have in an ideal monad $\mathbb{S}$ just a "restriction" $\mu' : S'S \longrightarrow S'$ of μ. It is natural to assume additionally that μ' obeys certain laws similar to the ones for the monad multiplication μ; this leads to the requirement that (S', μ') be an S-module, see Definition 5.5 below.

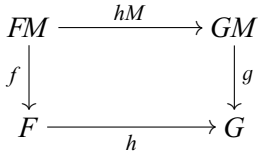
(ii) Another restriction in the previous section was the requirement that an ideal monad $\mathbb{S}$ should satisfy $S' + Id$ so that intuitively S' gives an abstract notion of "non-variables" to be used as the allowed right-hand sides of guarded equation morphisms. We have used that property of S in part (1) of the proof of Theorem 4.3. A different point of view is that of equipping a monad with some abstract notion of "allowed right-hand sides of equations", which leads us to the notion of idealized monad as introduced below.

Definition 5.2. Let (M, η, μ) be a monad on $\mathcal{A}$. A *(right) M-module* is a pair (F, f) consisting of an endofunctor F on $\mathcal{A}$ and a natural transformation $f : FM \longrightarrow F$ such that the following diagrams:



commute.

If (F, f) and (G, g) are M-modules, then a natural transformation $h : F \longrightarrow G$ such that the square



commutes is called a *module homomorphism.*

Remark 5.3. In [15], Section VII.4, (left) modules are defined under the name action for any monoidal category. The above Definition 5.2 states that definition for the special case of the monoidal category of endofunctors of $\mathcal{A}$ with composition as tensor product and the identity functor as unit. Here we chose the name module since in the monoidal category of abelian groups monoids are precisely rings and modules are the usual R-modules for a ring R.

Examples 5.4.

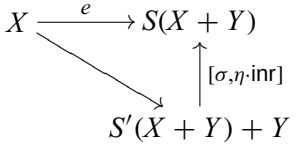
(i) Any monad (M, η, μ) is trivially an M-module (M, μ).

(ii) If $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ is an ideal monad in the sense of Definition 3.3, then (S', μ') is an S-module. This follows easily from the monad laws for S using the fact that the coproduct injections $\sigma_Y : S'Y \longrightarrow SY$ are monomorphic.

Definition 5.5. An *idealized monad* $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ consists of a monad (S, η, μ), an S-module (S', μ'), and a module homomorphism $\sigma : (S', \mu') \longrightarrow (S, \mu)$.

We call $\mathbb{S}$ *ideal* if $S = S' + Id$ with coproduct injections σ and η.

An idealized monad $\mathbb{S}$ is called *completely iterative* if any guarded equation morphism



has a unique solution $e^\dagger : X \longrightarrow SY$ (i.e., such that Diagram (4.1) commutes).

A *morphism of idealized monads* between $\mathbb{S}$ and $\mathbb{M} = (M, \eta^M, \mu^M, M', m, \mu'^M)$ is a pair (h, h') consisting of a monad morphism $h : (S, \eta, \mu) \longrightarrow (M, \eta^M, \mu^M)$ and a natural transformation $h' : S' \longrightarrow M'$ such that the squares

$$\begin{array}{ccc} S'S & \xrightarrow{h'*h} & M'M \\ {\scriptstyle \mu'}\downarrow & & \downarrow{\scriptstyle \mu'^M} \\ S' & \xrightarrow[h']{} & M' \end{array} \qquad \text{and} \qquad \begin{array}{ccc} S' & \xrightarrow{h'} & M' \\ {\scriptstyle \sigma}\downarrow & & \downarrow{\scriptstyle m} \\ S & \xrightarrow[h]{} & M \end{array}$$

commute. (Notice that the left-hand square means that h' is a module homomorphism with change of base h.)

Remark 5.6. Notice that idealized monad morphisms $(h, h') : \mathbb{S} \longrightarrow \mathbb{M}$ between ideal monads are determined by their second components. In fact, since the equations $S = S' + Id$ and $M = M' + Id$ hold, the two equation $m \cdot h' = h \cdot \sigma$ and $h \cdot \eta = \eta^M$ imply that $h = h' + Id$.

Examples 5.7.

(i) Any ideal monad in the sense of Definition 3.3 is an ideal monad in the sense of Definition 5.5.
(ii) The monad $\mathbb{T}$ given by the free completely iterative H-algebras is an ideal monad.
(iii) The free semigroup monad $X \longmapsto X^+$ together with S' assigning to X the set of words in X of length at least n, for some $n > 2$, is an idealized monad which is not ideal. It is trivial to check that the restriction of the monad multiplication to S' satisfies the necessary laws.
(iv) Let Σ be a signature and let Σ' be a subsignature of Σ. Then T_Σ together with $S' = H_{\Sigma'} T_\Sigma$ is an idealized monad. Note that S' assigns to a set Y all finite and infinite Σ-trees over Y whose root node is labelled by a symbol from Σ'. Once again, the laws of an idealized monad are easy to check, and this is another example which is not ideal whenever Σ' is a proper subsignature of Σ.

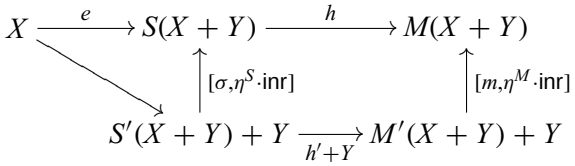
Notation 5.8. We denote by $\mathsf{CIzM}(\mathcal{A})$ the category of all idealized completely iterative monads and all idealized monad morphisms. By $\mathsf{CIM}(\mathcal{A})$ we denote its full subcategory consisting of all ideal completely iterative monads.

We also use $\mathsf{CIzAM}(\mathcal{A})$ to denote the full subcategory of $\mathsf{CIzM}(\mathcal{A})$ consisting of all *accessible* idealized monads $\mathbb{S}$, i.e., such that S and S' are accessible functors. Analogously $\mathsf{CIAM}(\mathcal{A})$.

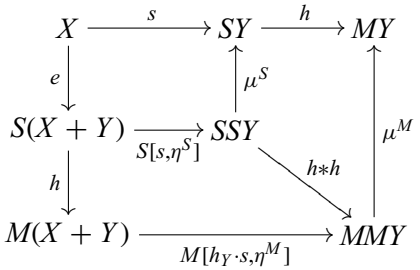
Proposition 5.9. *Idealized monad morphisms preserve solutions.*

Remark. More precisely, let $(h, h') : \mathbb{S} \longrightarrow \mathbb{M}$ be an idealized monad morphism between idealized monads. Then for every guarded equation morphism $e : X \longrightarrow S(X + Y)$ we get a guarded equation morphism $h_{X+Y} \cdot e$, and any solution s of e yields a solution $h_Y \cdot s$ of $h_{X+Y} \cdot e$. In particular, if $\mathbb{S}$ and $\mathbb{M}$ are completely iterative, we have $(h_{X+Y} \cdot e)^\dagger = h_Y \cdot e^\dagger$.

Proof. To see that $h_{X+Y} \cdot e$ is guarded, consider the commutative diagram



To see that $h_Y \cdot s$ solves $\overline{e}$, inspect the commutative diagram



where $*$ denotes parallel composition. □

Lemma 5.10. *If $\mathbb{S}$ is an idealized monad, then $S' + Id$ yields an ideal monad.*

Remark. By this we mean, of course, the sixtuple

$$\widetilde{\mathbb{S}} = (S' + Id, \widetilde{\eta}, \widetilde{\mu}, S', \mathsf{inl}, \widetilde{\mu}'),$$

where

$$\widetilde{\eta} \equiv Id \xrightarrow{\mathsf{inr}} S' + Id\,,$$
$$\widetilde{\mu}' \equiv S'(S' + Id) \xrightarrow{S'[\sigma,\eta]} S'S \xrightarrow{\mu'} S'\,,$$
$$\widetilde{\mu} \equiv (S' + Id)^2 = S'(S' + Id) + S' + Id \xrightarrow{[\widetilde{\mu}',S']+Id} S' + Id\,.$$

The proof of this result is essentially straightforward and involves only diagram chasing arguments using the axioms of the given idealized monad $\mathbb{S}$. For the sake of brevity we leave it to the reader. A very similar result was proved as Lemma 3.4 in [16].

Proposition 5.11. *The natural transformation $[\sigma, \eta] : S' + Id \longrightarrow S$ yields a morphism*

$$([\sigma, \eta], 1_{S'}) : \widetilde{\mathbb{S}} \longrightarrow \mathbb{S}$$

of idealized monads. And this is a coreflection of $\mathbb{S}$ in the category of ideal monads and morphisms of idealized monads.

Remark 5.12. More precisely, for any ideal monad

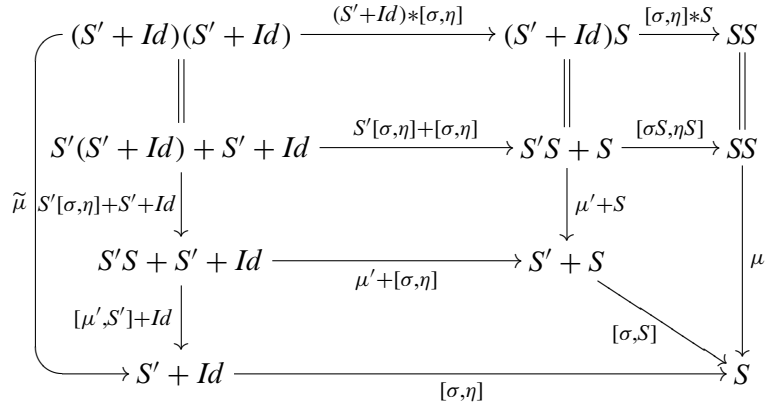
$$\mathbb{M} = (M, \eta^M, \mu^M, M', m, \mu'^M)$$

and any morphism of idealized monads $(h, h') : \mathbb{M} \longrightarrow \mathbb{S}$ there exists a unique idealized monad morphism $(\overline{h}, \overline{h}') : \mathbb{M} \longrightarrow \widetilde{\mathbb{S}}$ such that

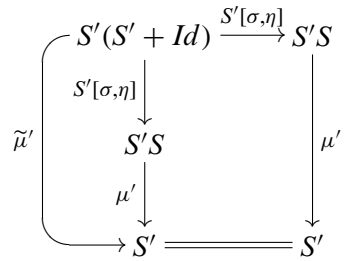
$$[\sigma, \eta] \cdot \overline{h} = h \quad \text{and} \quad \overline{h}' = h'. \tag{5.1}$$

That means that the inclusion of the full subcategory of ideal monads in the category of idealized monads has a right adjoint.

Proof. (1) We start by showing that $[\sigma, \eta] : S' + Id \longrightarrow S$ is a monad morphism. In fact, the unit law is obvious. For the associativity consider the commutative diagram

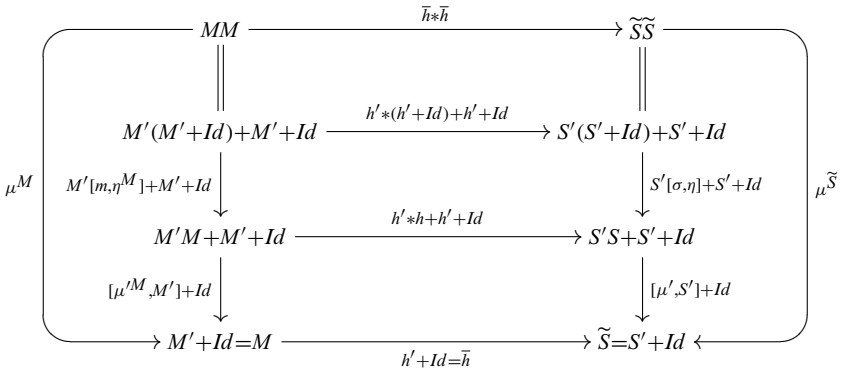


That $1_{S'}$ is a "restriction" of $[\sigma, \eta]$ is trivial, we have $[\sigma, \eta] \cdot \mathsf{inl} = \sigma = \sigma \cdot 1_{S'}$. Finally, $1_{S'}$ is a module homomorphism with change of base $[\sigma, \eta]$:



Thus, $([\sigma, \eta], 1_{S'})$ is a morphism of idealized monads.

(2) *Existence.* Put $\widetilde{S} = S' + Id$. Given $\mathbb{M}$ and (h, h'), then $(h' + Id, h') : \mathbb{M} \longrightarrow \widetilde{\mathbb{S}}$ is the desired ideal monad morphism. In fact, preservation of units is obvious. For the multiplication consider the following diagram



whose commutativity easily follows from axioms for ideal(ized) monads.

We leave the task to check that h' is a module homomorphism with change of base $h' + Id$, i.e., a restriction of h (see Definition 5.5), to the reader. This follows easily from the corresponding properties of (h, h').

Finally, we need to check the first equation of (5.1):

$$\begin{aligned}
[\sigma, \eta] \cdot \overline{h} &= [\sigma, \eta] \cdot (h' + Id) \\
&= [\sigma \cdot h', \eta] \\
&= [h \cdot m, h \cdot \eta^M] \\
&= h \cdot [m, \eta^M] \\
&= h.
\end{aligned}$$

(3) *Uniqueness.* Given any morphism of idealized monads $(\overline{h}, \overline{h}') : \mathbb{M} \longrightarrow \widetilde{S}$ satisfying (5.1), we immediately have $\overline{h}' = h'$, and therefore $\overline{h} = \overline{h}' + Id = h' + Id$. □

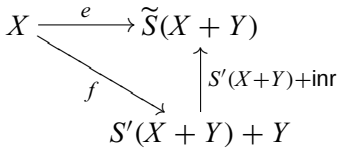
Lemma 5.13. *If $\mathbb{S}$ is a completely iterative monad, then so is its coreflection $\widetilde{\mathbb{S}}$.*

Remark. This result means, that the restriction $\mathsf{CIM}(\mathcal{A}) \longrightarrow \mathsf{CIzM}(\mathcal{A})$ of the embedding of Remark 5.12 also has a right adjoint. Notice also that this adjunction also clearly holds for the respective subcategories of accessible completely iterative monads:

$$\mathsf{CIAM}(\mathcal{A}) \underset{\longleftarrow}{\overset{\longrightarrow}{\perp}} \mathsf{CIzAM}(\mathcal{A}).$$

The proof of this result is similar to the proof of Lemma 3.5 in [16]. In our current setting it can be simplified due to Proposition 5.9.

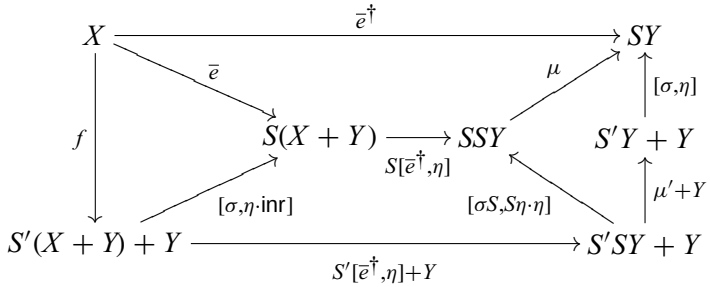
Proof. Denote for any object X by $\widetilde{S}X$ the coproduct $S'X + X$. We have to show that any guarded equation morphism



has a unique solution $e^\dagger : X \longrightarrow \widetilde{S}Y$. Define another guarded equation morphism w.r.t. $\mathbb{S}$ by composing with the coreflection arrow:

$$\overline{e} \equiv X \xrightarrow{e} \widetilde{S}(X + Y) \xrightarrow{[\sigma,\eta]} S(X + Y).$$

That $\overline{e}$ is indeed guarded follows from Proposition 5.9. Solve $\overline{e}$ to obtain a unique arrow $\overline{e}^\dagger : X \longrightarrow SY$ such that the upper part of the following diagram commutes:



(5.2)

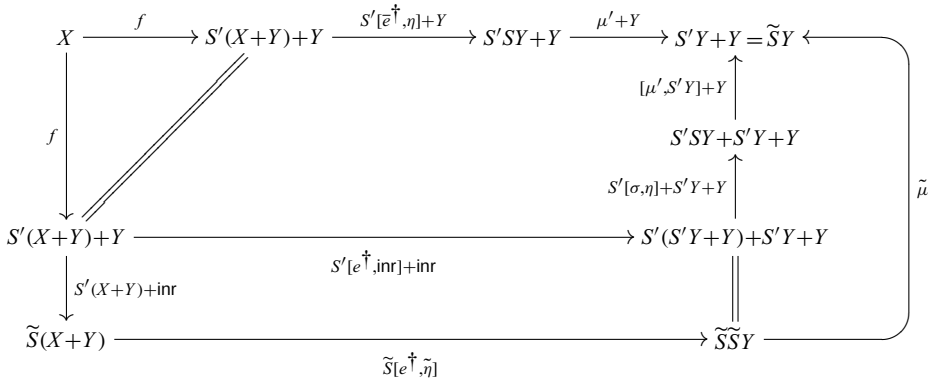
Then the outer square commutes, since the other three inner parts clearly do.

We shall prove that the following morphism

$$e^\dagger \equiv X \xrightarrow{f} S'(X+Y)+Y \xrightarrow{S'[\overline{e}^\dagger,\eta]+Y} S'SY+Y \xrightarrow{\mu'+Y} S'Y+Y = \widetilde{S}Y \tag{5.3}$$

is a unique solution of e.

That this morphism solves e follows from inspection of the diagram



It commutes except, perhaps, for the upper middle part, which we consider componentwise. The right-hand coproduct component is obvious, and for the left-hand one notice that the last arrow is μ' on both paths. We show that the rest already commutes, even when S' is removed, i.e., we plug in the definition (5.3) of $e^\dagger$ and obtain the commutative diagram:

$$\begin{array}{ccccc} X+Y & & \xrightarrow{\quad\quad [\overline{e}^\dagger,\eta] \quad\quad} & & SY \\ {\scriptstyle [f,\mathsf{inr}]}\downarrow & & & & \uparrow{\scriptstyle [\sigma,\eta]} \\ S'(X+Y)+Y & \xrightarrow[S'[\overline{e}^\dagger,\eta]+Y]{} & S'SY+Y & \xrightarrow[\mu'+Y]{} & S'Y+Y \end{array}$$

Its right-hand component is obvious, and the left-hand one is the outer square Diagram (5.2).

We have proved existence of a solution of e so far. As for the unicity suppose that $s : X \longrightarrow \widetilde{S}Y$ is any solution of e, i.e., the following diagram commutes:

$$\begin{array}{ccccc} X & \xrightarrow{\quad s \quad} & S'Y+Y=\widetilde{S}Y & & \\ & & \uparrow{\scriptstyle [\mu',S'Y]+Y} & & \\ {\scriptstyle f}\downarrow & & S'SY+S'Y+Y & & \\ & & \uparrow{\scriptstyle S'[\sigma,\eta]+S'Y+Y} & & \widetilde{\mu} \\ S'(X+Y)+Y & \xrightarrow[S'[s,\mathsf{inr}]+\mathsf{inr}]{} & S'(S'Y+Y)+S'Y+Y & & \\ {\scriptstyle [\mathsf{inl},\mathsf{inr}]}\downarrow & & \| & & \\ \widetilde{S}(X+Y) & & \xrightarrow[\widetilde{S}[s,\widetilde{\eta}]]{} & & \widetilde{S}\widetilde{S} \end{array} \tag{5.4}$$

Since $[\sigma,\eta] : \widetilde{S} \longrightarrow S$ is the first component of an idealized monad morphism (the coreflection arrow), the following morphism:

$$X \xrightarrow{\ s\ } S'Y+Y \xrightarrow{[\sigma,\eta]} SY \tag{5.5}$$

solves $\overline{e}$, see Proposition 5.9. Then it is not difficult to show that $s = e^\dagger$. In fact, start with the definition of the solution $e^\dagger$

$$e^\dagger = (\mu'_Y + Y) \cdot (S'[\overline{e}^\dagger, \eta_Y] + Y) \cdot f,$$

then substitute (5.5) for $\overline{e}^\dagger$ to obtain

$$(\mu'_Y + Y) \cdot (S'[[\sigma,\eta] \cdot s, \eta_Y] + Y) \cdot f, \tag{5.6}$$

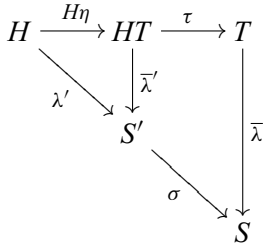
and finally use the equation $\eta_Y = [\sigma_Y, \eta_Y] \cdot \mathsf{inr}$ in order to see that (5.6) is the same as

$$(\mu'_Y + Y) \cdot (S'[\sigma,\eta] + Y) \cdot (S'[s,\mathsf{inr}] + Y) \cdot f,$$

which is just s due to the upper left-hand part of Diagram (5.4). □

At this point we are ready to prove the main result of this section, i.e., we extend the freeness result of Theorem 4.3 to all idealized monads. Thus, we establish that (b) implies (c) (see Section 1) in full generality.

Theorem 5.14. *For every completely iterative monad $\mathbb{S}$ and every ideal natural transformation λ: $H \longrightarrow S$ there exists a unique idealized monad morphism $(\overline{\lambda}, \overline{\lambda}') : \mathbb{T} \longrightarrow \mathbb{S}$ such that the following diagram:*



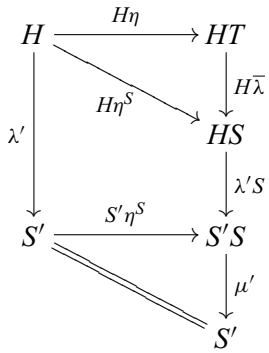
(5.7)

commutes.

Proof. First, suppose that $\mathbb{S}$ is an ideal monad. Theorem 4.3 states that the monad $\mathbb{T}$ of free cias is free w.r.t. all ideal completely iterative monads $\mathbb{S}$, whenever coproduct injections are monomorphic in $\mathcal{A}$. The same proof works in our current setting (i.e., with coproduct injections not necessarily monomorphic) with some minor modifications only. For the existence part we must verify that the induced pair $(\overline{\lambda}, \overline{\lambda}')$, where

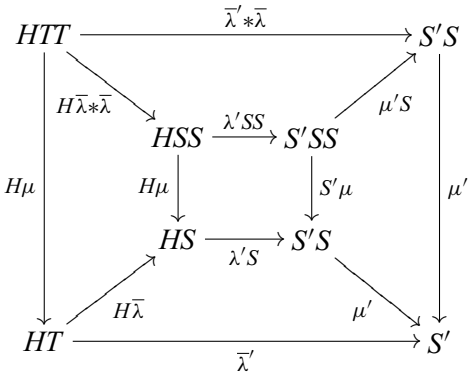
$$\overline{\lambda}' \equiv HT \xrightarrow{H\overline{\lambda}} HS \xrightarrow{\lambda' S} S'S \xrightarrow{\mu'} S', \tag{5.8}$$

is a morphism of idealized monads, and that the left-hand triangle in (5.7) commutes. For the latter, consider the following diagram



It commutes: for the upper triangle use the fact that $\overline{\lambda}$ is a monad morphism, for the middle part use the naturality of λ', and the lower triangle is the unit law of the $\mathbb{S}$-module S'.

To see that $(\overline{\lambda}, \overline{\lambda}')$ is an idealized monad morphism, it only remains to show that $\overline{\lambda}'$ is a module homomorphism with change of base $\overline{\lambda}$. Consider the following diagram

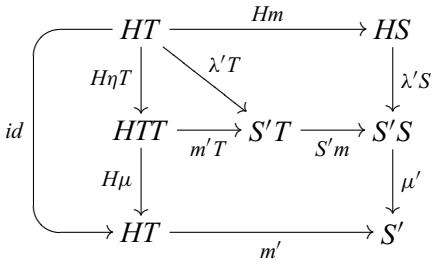


For the upper part it suffices to consider the parallel components for HT and T separately. The HT-part is (5.8), the other one is trivial. The other parts of the diagram are clear. For the middle three parts use—from left to right—that $\overline{\lambda}$ is a monad morphism, naturality of λ' and the module laws for S', and the lower part is (5.8) again.

For the uniqueness of $(\overline{\lambda}, \overline{\lambda}')$ assume that $(m, m') : \mathbb{T} \longrightarrow \mathbb{S}$ is an idealized monad morphism such that (5.7) with $(\overline{\lambda}, \overline{\lambda}')$ replaced by (m, m') commutes. Then from the proof of Theorem 4.3 we conclude that $m = \overline{\lambda}$ and from this it follows that:

$$m' = \mu' \cdot \lambda' S \cdot Hm = \mu' \cdot \lambda' S \cdot H\overline{\lambda} = \overline{\lambda}'.$$

In fact, to see the first equality consider the diagram



The lower square commutes since m' is a module homomorphism with change of base m, the left-hand part does by the unit law of the monad $\mathbb{T}$, the upper triangle by (5.7) and the upper right-hand part by naturality of λ'.

We have established the desired result for all ideal completely iterative monads $\mathbb{S}$. Now if $\mathbb{S}$ is an arbitrary (idealized) completely iterative monad, form its ideal coreflection $\widetilde{\mathbb{S}}$, which is completely iterative by Lemma 5.13. We have an ideal transformation

$$t \equiv H \xrightarrow{\lambda'} S' \xrightarrow{\mathsf{inl}} \widetilde{S}$$

inducing a unique morphism of idealized monads $(\overline{t}, \overline{t}') : \mathbb{T} \longrightarrow \widetilde{\mathbb{S}}$ which extends t. Use the adjunction of Lemma 5.13 to see that the composition of $(\overline{t}, \overline{t}')$ with the coreflection arrow $\widetilde{\mathbb{S}} \longrightarrow \mathbb{S}$ yields the desired unique idealized monad morphism extending the given ideal transformation λ.

Remark 5.15. For accessible functors the last part of the proof just composes the two adjunctions from Remark 4.4 and Lemma 5.13 (see 5.8 for notation)

$$\mathsf{CIzAM}(\mathcal{A}) \underset{\longrightarrow}{\overset{\longleftarrow}{\perp}} \mathsf{CIAM}(\mathcal{A}) \underset{\longrightarrow}{\overset{\longleftarrow}{\perp}} \mathsf{Acc}[\mathcal{A}, \mathcal{A}]$$

and it is clear that this extends to all completely iterative monads using the freeness of $\mathbb{T}$ for ideal completely iterative monads and (the full strength of) the adjunction from Lemma 5.13.

For the record we note the following result.

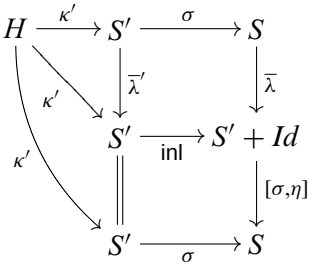
Proposition 5.16. *If $\mathbb{S}$ is a free completely iterative monad on H then it is ideal.*

Proof. We will show that the ideal coreflection $\widetilde{\mathbb{S}} \longrightarrow \mathbb{S}$ is an isomorphism. Recall that $\widetilde{\mathbb{S}}$ is given by $\widetilde{S} = S' + Id$. Since $\mathbb{S}$ is completely iterative, so is $\widetilde{\mathbb{S}}$ by Lemma 5.13. Moreover, from the universal arrow $\kappa : H \longrightarrow S$ we get an ideal transformation

$$\lambda \equiv H \xrightarrow{\kappa'} S' \xrightarrow{\mathsf{inl}} S' + Id.$$

Thus, by the freeness of $\mathbb{S}$ we have a unique idealized monad morphism $\alpha = (\overline{\lambda}, \overline{\lambda}') : \mathbb{S} \longrightarrow \widetilde{\mathbb{S}}$ extending λ. We shall show that this is an inverse of the coreflection arrow $\beta = ([\sigma, \eta], 1_{S'}) : \widetilde{\mathbb{S}} \longrightarrow \mathbb{S}$.

(i) $\beta \cdot \alpha = 1_{\mathbb{S}}$: Consider the following commutative diagram:



It shows that $\beta \cdot \alpha$ is an idealized monad morphism extending $\kappa = \sigma \cdot \kappa'$, whence it must be the identity on $\mathbb{S}$.
(ii) $\alpha \cdot \beta = 1_{\widetilde{\mathbb{S}}}$: From $\beta \cdot \alpha = 1_{\mathbb{S}}$ we get in the second component $\overline{\lambda}' = 1_{S'}$. Hence, we must only check the first component of $\alpha \cdot \beta$. Consider the coproduct components of $S' + Id$ separately to see that $\overline{\lambda} \cdot [\sigma, \eta] = [\mathsf{inl} \cdot \overline{\lambda}', \widetilde{\eta}] = [\mathsf{inl}, \mathsf{inr}] = 1_{S'+Id}$. □

6. Iterability is necessary

In this section, we assume that $\mathcal{A}$ is a category with binary coproducts. We have seen above that any iteratable endofunctor on $\mathcal{A}$ admits a free completely iterative monad. In this section, we prove that, conversely, every endofunctor admitting a free completely iterative monad is iteratable. This is a new result, which has only appeared in the extended abstract [16]. It is the last ingredient we need

to complete the main task of the current paper, i.e., to establish that the statements (a), (b), and (c) from the introduction are equivalent. Fortunately, as compared to [16] the proof is now relatively short since all the necessary technical auxiliary results have already been established in Section 5. Also because of the equivalence of statements (a) and (b), the proof can be somewhat simplified.

Theorem 6.1. *Every endofunctor generating a free completely iterative monad is iteratable.*

Remark 6.2. More detailed, suppose that H is an endofunctor on $\mathcal{A}$ and

$$\kappa : H \longrightarrow S$$

is a free completely iterative monad on H (where κ is an ideal transformation), then for all objects Y of $\mathcal{A}$, SY is a free completely iterative H-algebra on Y with universal arrow $\eta_Y : Y \longrightarrow SY$, and it follows that H is iteratable, see Corollary 2.11.

Proof. Let a free completely iterative monad $\mathbb{S} = (S, \eta, \mu, S', \sigma, \mu')$ on H with universal arrow κ: $H \longrightarrow S$ be given. Observe first that $(HS, H\mu)$ forms a right S-module. In fact, the module laws follow trivially from the monad laws for S. The following natural tranformation:

$$s \equiv HS \xrightarrow{\kappa S} SS \xrightarrow{\mu} S$$

is a module homomorphism $(HS, H\mu) \longrightarrow (S, \mu)$. To see this inspect the commutative diagram

$$\begin{array}{ccccc} HSS & \xrightarrow{\kappa SS} & SSS & \xrightarrow{\mu S} & SS \\ {\scriptstyle H\mu}\downarrow & & {\scriptstyle S\mu}\downarrow & & \downarrow{\scriptstyle \mu} \\ HS & \xrightarrow[\kappa S]{} & SS & \xrightarrow[\mu]{} & S \end{array}$$

Thus, we have an idealized monad

$$\overline{\mathbb{S}} = (S, \eta, \mu, HS, s, H\mu).$$

This monad is completely iterative, since any guarded equation morphism e for $\overline{\mathbb{S}}$ is also guarded for $\mathbb{S}$. To see this consider the commutative diagram

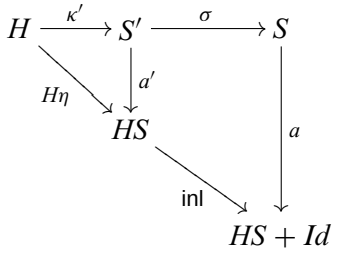
$$\begin{array}{ccc} X & \xrightarrow{\quad e \quad} & S(X+Y) \\ & \searrow \quad \nearrow_{[s,\eta\cdot\mathsf{inr}]} & \uparrow{\scriptstyle [\sigma,\eta\cdot\mathsf{inr}]} \\ & HS(X+Y)+Y & \xrightarrow[\mu'\cdot\kappa' S+Y]{} S'(X+Y)+Y \end{array}$$

Thus, e has a unique solution.

Now denote by $\widetilde{\mathbb{S}}$ the ideal coreflection of $\overline{\mathbb{S}}$ whose underlying functor is given by $HS + Id$. We clearly have an ideal transformation

$$\lambda \equiv H \xrightarrow{H\eta} HS \xrightarrow{\mathsf{inl}} HS + Id.$$

Since $\mathbb{S}$ is free on H, we obtain a unique idealized monad morphism $\alpha = (a, a') : \mathbb{S} \longrightarrow \widetilde{\mathbb{S}}$ extending λ, i.e., such that the diagram

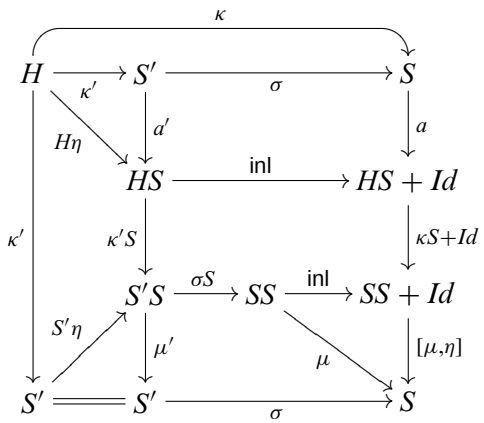


commutes. Let us prove that α is an isomorphism with an inverse given by

$$\beta = (b, b') \equiv \widetilde{\mathbb{S}} \xrightarrow{([\mu\cdot\kappa S,\eta],1_{HS})} \overline{\mathbb{S}} \xrightarrow{(1_S,\mu'\cdot\kappa' S)} \mathbb{S}$$

It is not difficult to see that β is an idealized monad morphism. The first morphism is the coreflection arrow, and for the second one it is clear that 1_S is a monad morphism and it is easy to see that $\mu' \cdot \kappa' S$: $HS \longrightarrow S'S \longrightarrow S'$ is a module homomorphism (with change of base 1_S, i.e., no change of base).

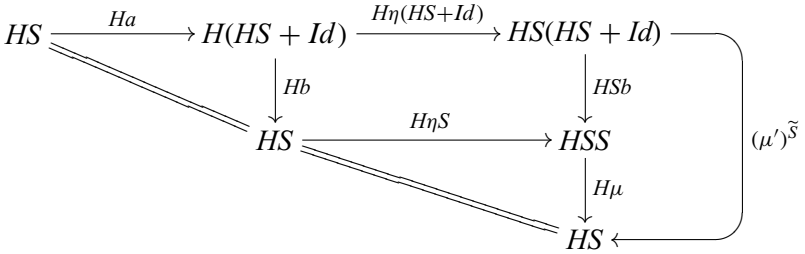
(1) $\beta \cdot \alpha = 1_{\mathbb{S}}$: Notice that $\beta \cdot \alpha$ is an idealized monad morphism extending the universal arrow κ, i.e., the following diagram



commutes. By the freeness of $\mathbb{S}$, $\beta \cdot \alpha$ must be the identity on $\mathbb{S}$.
(2) $\alpha \cdot \beta = 1_{\widetilde{\mathbb{S}}}$: Since $\widetilde{\mathbb{S}}$ is an ideal monad, it suffices to check the second component of $\alpha \cdot \beta$ (see Remark 5.6). Hence, we show that $a' \cdot b' = 1_{HS}$:

$$\begin{aligned} a' \cdot b' &= a' \cdot \mu' \cdot \kappa' S && \text{(definition of } b') \\ &= (\mu')^{\widetilde{S}} \cdot (a' * a) \cdot \kappa' S && (a' \text{ is a module homomorphism}) \\ &= (\mu')^{\widetilde{S}} \cdot (H\eta * a) && (a' \cdot \kappa' S = H\eta), \end{aligned}$$

where $*$ denotes parallel composition. Analyzing the last arrow further, we finally get the following commutative diagram



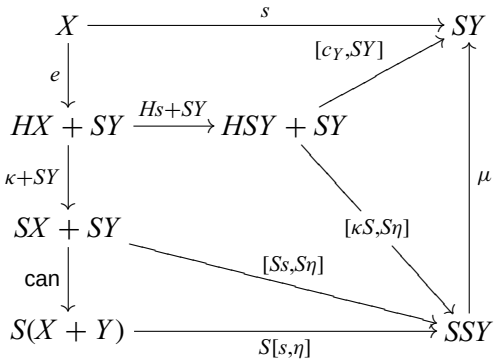
For the left-hand triangle use that $\beta \cdot \alpha = 1_{\mathbb{S}}$, the lower triangle is one of the monad laws of S, the upper square is naturality of η, and the right-hand part is the definition of μ' for the ideal coreflection $\widetilde{\mathbb{S}}$ (see Lemma 5.10). Thus $a' \cdot b' = 1_{HS}$, and therefore $\alpha \cdot \beta$ is the identity on $\widetilde{\mathbb{S}}$ as desired.

To complete the proof we show now that SY carries the structure of a free cia on Y. In fact, notice first that $b_Y \cdot \mathsf{inr} = \eta_Y$. We shall prove below that SY with the structure map $c_Y = b_Y \cdot \mathsf{inl} = \mu_Y \cdot \kappa_{SY}$ is a completely iterative H-algebra. Then, it follows from the proof of Theorem 2.10 that (SY, b_Y) is a completely iterative algebra of $H(\_) + Y$, and since b_Y is an isomorphism this cia is initial, see Remark 2.9. Thus, by Theorem 2.10, (SY, c_Y) is a free cia with universal arrow $\eta_Y : Y \longrightarrow SY$.

Now in order to see that (SY, c_Y) is a cia let $e : X \longrightarrow HX + SY$ be a flat equation morphism. Then form the following equation morphism:

$$\overline{e} \equiv X \xrightarrow{e} HX + SY \xrightarrow{\kappa + SY} SX + SY \xrightarrow{\mathsf{can}} S(X + Y)$$

for the monad $\mathbb{S}$. Since κ is an ideal transformation, $\overline{e}$ is guarded. Solutions of $\overline{e}$ w.r.t. the completely iterative monad $\mathbb{S}$ are in one-to-one correspondence with solutions of e w.r.t. the algebra (SY, c_Y). Indeed, consider the diagram



The arrow s is a solution of $\overline{e}$ if and only if the outer square of the diagram commutes. Equivalently, the upper part commutes since all other parts obviously do. But this is precisely the case if s solves e. Thus, since $\overline{e}$ has a unique solution so does e. □

To conclude the paper let us collect the results of Corollary 2.11, and Theorems 5.14 and 6.1 to state our main result compactly.

Corollary 6.3. *Let H be an endofunctor on $\mathcal{A}$, and let T be an object assignment of $\mathcal{A}$. Then the following are equivalent:*

(i) for every object Y of $\mathcal{A}$, TY is a final coalgebra of $H(\_) + Y$, i.e., H is iteratable,
(ii) for every object Y of $\mathcal{A}$, TY is a free completely iterative algebra of H on Y, and
(iii) T is a free completely iterative monad on H.

Acknowledgments

I am deeply indebted to Jiří Adámek and Jiří Velebil. Their comments and support helped a lot to bring this paper to its current form. I also thank the anonymous referee for his comments.

References

[1] P. Aczel, J. Adámek, S. Milius, J. Velebil, Infinite trees and completely iterative theories: a coalgebraic view, Theoret. Comput. Sci. 300 (2003) 1–45.
[2] J. Adámek, S. Milius, J. Velebil, Free iterative theories—a coalgebraic view, Math. Struct. Comp. Sci. 13 (2003) 259–320.
[3] J. Adámek, S. Milius, J. Velebil, On rational monads and free iterative theories, Electron. Notes Theor. Comput. Sci. 69 (2003).
[4] J. Adámek, S. Milius, J. Velebil, From iterative algebras to iterative theories, preprint, 2004, extended abstract to appear in Electron. Notes Theor. Comput. Sci.
[5] J. Adámek, S. Milius, J. Velebil, On iterative algebras and Elgot algebras, 2004 (submitted).
[6] P. America, J.J.M.M. Rutten, Solving reflexive domain equations in a category of complete metric spaces, J. Comp. Syst. Sci. 39 (1989) 343–375.
[7] J. Barwise, L.S. Moss, Vicious Circles, CSLI Public. Notes, vol. 60, Stanford, 1996.
[8] B. Courcelle, Fundamental properties of infinite trees, Theoret. Comput. Sci. 25 (1983) 95–169.
[9] C.C. Elgot, Monadic computation and iterative algebraic theories, in: H.E. Rose, J.C. Shepherdson (Eds.), Logic Colloquium '73, North-Holland Publishers, Amsterdam, 1975.
[10] C.C. Elgot, S.L. Bloom, R. Tindell, On the algebraic structure of rooted trees, J. Comp. Syst. Sci. 16 (1978) 361–399.
[11] N. Ghani, T. Uustalu, Coproducts of ideal monads, Theor. Inform. Appl. 38 (4) (2004) 321–342.
[12] J. Lambek, A fixpoint theorem for complete categories, Math. Z. 103 (1968) 151–161.
[13] F.W. Lawvere, Functorial semantics of algebraic theories, dissertation, Columbia University, 1963.
[14] F.E.J. Linton, Some aspects of equational categories, in: Proceedings of the Conference on Categorical Algebra, Springer, Berlin, 1966, pp. 84–94.
[15] S. Mac Lane, Categories for the Working Mathematician, second ed., Springer, Berlin, 1998.
[16] S. Milius, On iteratable endofunctors, Electron. Notes Theor. Comput. Sci. 69 (2003).
[17] L.S. Moss, Parametric corecursion, Theoret. Comput. Sci. 260 (1–2) (2001) 139–163.
[18] E. Nelson, Iterative algebras, Theoret. Comput. Sci. 25 (1983) 67–94.
[19] G. Osius, Categorical set theory: a characterization of the category of sets, J. Pure Appl. Algebra 4 (1974) 79–119.
[20] J. Tiuryn, Unique fixed points vs. least fixed points, Theoret. Comput. Sci. 12 (1980) 229–254.
[21] J. Worrell, On the final sequence of a finitary set functor, Theoret. Comput. Sci. (accepted).

Electronic Notes in Theoretical Computer Science 69 (2003)
URL: http://www.elsevier.nl/locate/entcs/volume69.html 18 pages

On Iteratable Endofunctors

Stefan Milius [1]

Institute of Theoretical Computer Science, Technical University, Braunschweig, Germany

Abstract

Completely iterative monads of Elgot et al. are the monads such that every guarded iterative equation has a unique solution. Free completely iterative monads are known to exist on every iteratable endofunctor H, i. e., one with final coalgebras of all functors $H(\_) + X$. We show that conversely, if H generates a free completely iterative monad, then it is iteratable.

Key words: monad, completely iterative, iterable

1 Introduction

There have been various attempts to algebraically capture the concept of computations on data through a program, taking into account that such computations are potentially infinite. During the 1970's the ADJ-group studied continuous algebras, i. e., algebras endowed with a CPO structure. There, an infinite computation is a join of the directed set of its finite approximations, see e. g. [ADJ]. Later, algebras over complete metric spaces were considered, where an infinite computation is a limit of a Cauchy sequence of finite approximations, see e. g. [ARu].

Another approach to infinite computations are iterative algebraic theories, introduced by Calvin C. Elgot in [E]. This notion has been extended to the notion of completely iterative theories by Elgot, Bloom and Tindell, see [EBT]. The latter are algebraic theories (in the sense of Lawvere and Linton [Lin]) that allow for unique solutions of fixed point equations. An important example of a completely iterative theory is the theory of finite and infinite trees over a given signature Σ. In [EBT] it is shown that this is the free completely iterative theory over Σ.

It has recently been discovered by Peter Aczel, Jiří Adámek, Jiří Velebil, and the present author [AAMV], that the above fact is a special case of a much more general categorical result using a coalgebraic approach to infinite

[1] Email: milius@iti.cs.tu-bs.de

computation. This coalgebraic approach has also independently been studied by Larry Moss in [M]. Here one considers a category $\mathcal{A}$ with binary coproducts, and an *iteratable* endofunctor H on $\mathcal{A}$, i. e., such that for every object X a final coalgebra

$$TX$$

of $H(\_) + X$ exists. In [AAMV] the notion of a completely iterative monad is introduced. Informally, this is a monad that allows for unique solutions of systems of equations of a certain liberal type. It has been shown that the mapping $X \mapsto TX$ is the object assignment of a completely iterative monad. Moreover, it was proved that this monad T is a free completely iterative monad on H.

In the present paper we investigate the exact relationship between the notion of iteratability and the existence of free completely iterative monads for an endofunctor. The main result of [AAMV] shows that iteratable endofunctors admit free completely iterative monads. Here we prove that no other functors do so. More precisely, if S is a free completely iterative monad over an endofunctor H on $\mathcal{A}$, then H is iteratable, and for all objects X of $\mathcal{A}$, SX is a final coalgebra of $H(\_) + X$.

Before we prove our main result in Section 3 we shall recall the results of [AAMV] and give some motivation for the notion of completely iterative monad in Section 2.

2 Iteratable Endofunctors and Completely Iterative Monads

2.1 A Motivating Example

We take from [AMV] a motivating example for the coalgebraic approach of [AAMV]. Consider the algebra of finite and infinite trees over a given signature Σ. This algebra allows for the unique solution of systems of so-called guarded equations. Let us give the details of this. Denote by

$$T_\Sigma X$$

the algebra of all finite and infinite Σ-labelled trees with variables from X. That is, trees labelled so that a node with $n > 0$ children is labelled by an n-ary operation symbol (an element of Σ_n) and a leaf is labelled by a variable or a constant (an element of $X + \Sigma_0$). The operations on $T_\Sigma X$ are given by tree-tupling. Furthermore, consider a system of equations

$$\begin{aligned} x_0 &\approx t_0(x_0, x_1, x_2, \ldots, y_0, y_1, y_2, \ldots) \\ x_1 &\approx t_1(x_0, x_1, x_2, \ldots, y_0, y_1, y_2, \ldots) \\ &\vdots \\ x_n &\approx t_n(x_0, x_1, x_2, \ldots, y_0, y_1, y_2, \ldots) \\ &\vdots \end{aligned} \tag{1}$$

where t_i are trees with variables from $X = \{x_0, x_1, x_2, \ldots\}$ and parameters from $Y = \{y_0, y_1, y_2, \ldots\}$, i.e.,

$$t_i \in T_\Sigma(X + Y) \quad \text{for } i = 0, 1, 2, \ldots$$

Notice that in a system we denote by $\approx$ formal equations and $=$ is the identity of the two sides. A system is called *guarded* provided that none of the trees t_i is just a variable from X. This condition is enough to force the existence of a unique *solution* of (1), i.e., a unique tuple ${x_i}^\dagger(y_0, y_1, y_2, \ldots)$ of trees in $T_\Sigma Y$ such that the identities

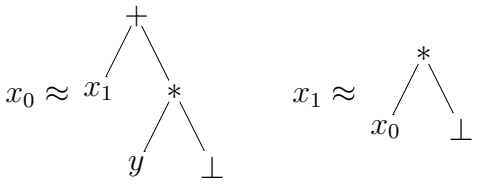
$$\begin{aligned}
{x_0}^\dagger &= t_0({x_0}^\dagger, {x_1}^\dagger, {x_2}^\dagger, \ldots, y_0, y_1, y_2, \ldots) \\
{x_1}^\dagger &= t_1({x_0}^\dagger, {x_1}^\dagger, {x_2}^\dagger, \ldots, y_0, y_1, y_2, \ldots) \\
&\vdots \\
{x_n}^\dagger &= t_n({x_0}^\dagger, {x_1}^\dagger, {x_2}^\dagger, \ldots, y_0, y_1, y_2, \ldots) \\
&\vdots
\end{aligned}$$

hold.

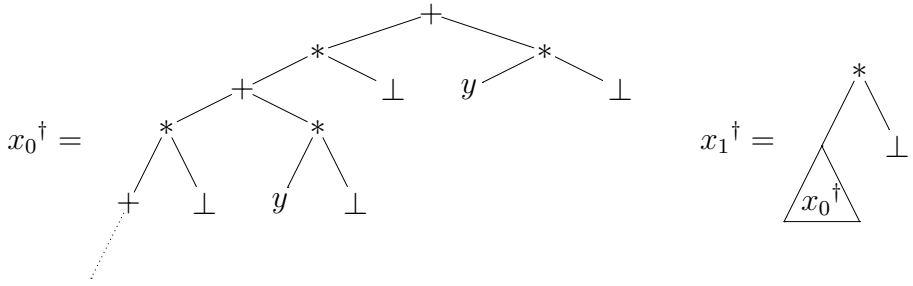
Theorem 2.1 *Every guarded system of equations has a unique solution.*

In fact, this is a special case of a much more general Solution Theorem we mention in Subsection 2.2 below.

Example 2.2 Let Σ consist of binary operations $+$ and $*$ and a constant $\bot$. The following system of equations



is guarded. The solution is given by the following trees in $T_\Sigma Y$:



2.2 Substitutions and Solutions Coalgebraically

The coalgebraic approach of [AAMV] and [M] relies on the following observation. To any signature Σ there is an associated polynomial endofunctor $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$ defined by

$$H_\Sigma X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X^2 + \cdots$$

Recall that H_Σ-algebras are just the classical universal Σ-algebras. A final H_Σ-coalgebra is well-known to be the coalgebra $T_\Sigma\emptyset$ of all finite and infinite Σ-labelled trees without variables, see [AK]. Now $H_\Sigma(\_)+X$ is also polynomial (for the signature obtained from Σ by adding a constant symbol for every element of X), thus, $T_\Sigma X$ is a final coalgebra of $H_\Sigma(\_) + X$.

Taking the existence of such a parametrized family of final coalgebras as the primitive notion, one can abstract away from signatures (=polynomial endofunctors) and from the category Set.

Assumption 2.3 *For the rest of this section we assume that $\mathcal{A}$ denotes a category with binary coproducts whose injections are monomorphic, and H is an endofunctor on $\mathcal{A}$.*

Definition 2.4 An endofunctor H of $\mathcal{A}$ is called *iteratable* if for every object X of $\mathcal{A}$ there exists a final coalgebra of $H(\_) + X$.

The following examples of iteratable endofunctors have been taken from [AAMV].

Example 2.5

(i) Accessible (=bounded) endofunctors on Set. An endofunctor is called accessible if it preserves λ-filtered colimits for some infinite cardinal λ. In [AP], it was shown that those are precisely the so-called bounded endofunctors. This example subsumes all the following ones.

(ii) (Generalized) polynomial endofunctors on Set, i. e., H is defined by

$$HZ = \coprod_{i<\lambda} A_i \times Z^i$$

for some cardinal λ; for $\lambda = \omega$ one has a polynomial endofunctor associated to a finitary signature as in 2.1.

(iii) The bounded power set functors defined on objects by

$$\mathcal{P}_\lambda X = \{Y \subseteq X \mid |Y| < \lambda\}$$

for some cardinal λ. Notice that the (unbounded) power set functor $\mathcal{P} : \mathsf{Set} \longrightarrow \mathsf{Set}$ does not allow for a final coalgebra, and hence, it is not iteratable.

Note that the notions of accessibility and iteratability are not equivalent. In fact, there are examples of non-accessible endofunctors that are iteratable (see [AAMV]).

Remark 2.6 If H is an iteratable endofunctor on $\mathcal{A}$ we denote by

$$TX$$

the final coalgebra of $H(\_)+X$. By the Lambek Lemma (see [L]), the structure map of that final coalgebra is an isomorphism, and consequently, TX is a

coproduct of HTX and X with injections

$$\eta_X : X \longrightarrow TX \qquad \text{“injection of variables”}$$
$$\tau_X : HTX \longrightarrow TX \qquad \text{“}TX \text{ is an } H\text{-algebra”}$$

The final coalgebras TX have a rich structure. Firstly, the way how substitution works on trees in $T_\Sigma X$ generalizes smoothly to the categorical setting. Recall here that given an interpretation of variables $x \in X$ as trees $s(x)$ over Y, i. e., a function $s : X \longrightarrow T_\Sigma Y$, then the corresponding substitution of trees from $T_\Sigma Y$ into (leaves of) trees of $T_\Sigma X$ is a homomorphism

$$\widehat{s} : T_\Sigma X \longrightarrow T_\Sigma Y$$

of Σ-algebras. Moreover, $\widehat{s}$ is the unique extension of s. This can be generalized to all iteratable endofunctors:

Substitution Theorem 2.7 *For any arrow $s : X \longrightarrow TY$ there exists a unique homomorphism $\widehat{s} : TX \longrightarrow TY$ of H-algebras extending s, i. e., such that $\widehat{s} \cdot \eta_X = s$.*

The proof can be found in [M] or [AAV] (slightly improved in [AAMV]).

Next, one can generalize in a straightforward way the notion of a system of equations. An *equation arrow* is a morphism

$$e : X \longrightarrow T(X + Y)$$

in $\mathcal{A}$. It is called *guarded* if it factors as follows

$$\begin{array}{ccc} X & \xrightarrow{\quad e \quad} & T(X+Y) \\ & \searrow & \uparrow{\scriptstyle [\tau_{X+Y},\, \eta_{X+Y} \cdot \mathsf{inr}]} \\ & & HT(X+Y) + Y\,. \end{array}$$

Notice that for a polynomial endofunctor $H = H_\Sigma$ on Set this is precisely the notion of a guarded system as presented above, since $T(X+Y) = HT(X+Y) + X + Y$. Finally, a *solution* for an equation arrow e is an arrow $e^\dagger : X \longrightarrow TY$ such that the following triangle

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & TY \\ {\scriptstyle e}\downarrow & \nearrow_{\widehat{[e^\dagger, \eta_Y]}} & \\ T(X+Y) & & \end{array}$$

commutes. Again, this corresponds precisely to the notion of solution for systems of equations in case of polynomial endofunctors on Set.

The following result is called Parametric Corecursion in [M] and Solution Theorem in [AAV]; see also an improved version of the proof in [AAMV]:

Solution Theorem 2.8 *Given an iteratable endofunctor H, every guarded equation morphism has a unique solution.*

Remark 2.9 It is an easy consequence of the Substitution Theorem that $(T, \eta, \widehat{(\_)})$ forms a Kleisli triple, i. e., the following three conditions are satisfied

(i) $\widehat{\eta_X} = id_{TX}$ for all objects X,

(ii) $\widehat{s} \cdot \eta_X = s$ for all arrows $s : X \longrightarrow TY$,

(iii) $\widehat{r} \cdot \widehat{s} = \widehat{\widehat{r} \cdot s}$ for any morhisms $s : X \longrightarrow TY$ and $r : Y \longrightarrow TZ$.

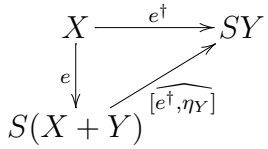
Thus setting $\mu_X = \widehat{id_{TX}} : TTX \longrightarrow TX$ we obtain a monad (T, η, μ), and we call it the *completely iterative monad* generated by H.

2.3 The free Completely Iterative Monad

Based on the consideration in the previous section 2.2 it is quite natural to call for any monad (S, η, μ) on $\mathcal{A}$ a morphism

$$e : X \longrightarrow S(X + Y)$$

an *equation arrow*. Recall that for any monad there is an associated Kleisli triple, where for $s : X \longrightarrow SY$ we have $\widehat{s} = \mu_Y \cdot Ss$. Hence, a morphism $e^\dagger : X \longrightarrow SY$ with

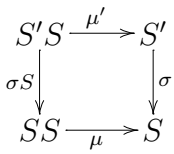


will be called a *solution*. However, it is in general not obvious how the property of e being guarded is to be expressed for an arbitrary monad.

Elgot, Bloom and Tindell [EBT] use, in their setting of algebraic theories, the notion of an ideal theory introduced by Elgot in [E]. For finitary monads on Set this notion is equivalent to the following notion of ideal monad (see [AAMV] for a simple proof of this fact):

Definition 2.10

(i) Let (S, η, μ) be a monad. A (right) ideal of S is a subfunctor $\sigma : S' \rightarrowtail S$ such that there exists a (necessarily unique) restriction $\mu' : S'S \longrightarrow S'$ of μ, i. e., the following square



commutes.

(ii) A monad together with an ideal of it is called an *idealized monad*. If furthermore we have $S = S' + Id$, i. e., $[\sigma, \eta] : S' + Id \longrightarrow S$ is an

isomorphism, then S is called an *ideal monad*.

(iii) An *idealized-monad morphism* between idealized monads S_1 and S_2 with chosen ideals $\sigma_i : S_i' \rightarrowtail S_i$, $i = 1, 2$, is a monad morphism $h : S_1 \longrightarrow S_2$ that preserves the chosen ideals, i. e., there exists a (necessarily unique) natural transformation $h' : S_1' \longrightarrow S_2'$ such that the following square

$$\begin{array}{ccc} S_1' & \xrightarrow{h'} & S_2' \\ {\scriptstyle \sigma_1}\downarrow & & \downarrow{\scriptstyle \sigma_2} \\ S_1 & \xrightarrow[h]{} & S_2 \end{array}$$

commutes.

Example 2.11

(i) Recall that the monad T is a coproduct of HT and Id. Hence the ideal $\tau : HT \rightarrowtail T$, where μ' is given by $H\mu$ makes T into an ideal monad.

(ii) Any monad S has ideals, e. g., the largest one (S itself). If $\mathcal{A}$ has a strict initial object, then the smallest ideal is given by the constant functor on the initial object.

Remark 2.12

(i) Notice that the notion of an ideal of a monad corresponds precisely to the notion of a right ideal for a monoid. Indeed, recall that a right ideal of a monoid M is a subset I of M such that $I \cdot M \subseteq I$. Now a monad is just a monoid in the monoidal category $[\mathcal{A}, \mathcal{A}]$ of endofunctors on $\mathcal{A}$ with tensor product being given by composition of functors.

(ii) It is not difficult to show that the category of ideal monads and ideal monad homomorphisms is a coreflective subcategory of the category of idealized monads with the same morphisms. In fact, if (S, η, μ) is a monad with ideal $\sigma : S' \rightarrowtail S$ the coreflection arrow is given by

$$S' + Id \xrightarrow{[\sigma, \eta]} S.$$

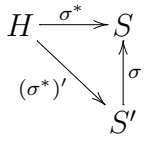
Since this is not needed here, the proof is omitted.

Remark 2.13 Observe that the completely iterative monad T generated by H comes with a natural "embedding of H"

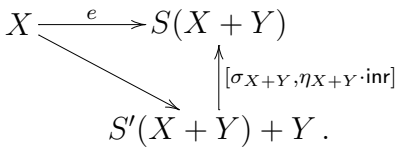
$$\tau^* \equiv H \xrightarrow{H\eta} HT \xrightarrow{\tau} T$$

into it. More generally, we call for any endofunctor H and idealized monad S a natural transformation $\sigma^* : H \longrightarrow S$ *ideal* if it factors through the ideal

$\sigma : S' \rightarrowtail S$ as follows



For an idealized monad S we define the notion of a *guarded* equation arrow as a morphism e that factors

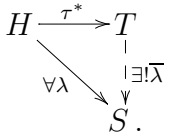


Definition 2.14 An idealized monad S is called *completely iterative* if every guarded equation arrow has a unique solution.

Remark 2.15 In [AAMV] a completely iterative monad is required to be ideal. Observe however, that this is an uneccessary restriction. In fact, all of the proofs of [AAMV] use only properties of idealized monads.

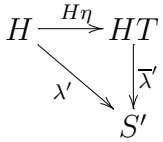
The following is the main result of [AAMV].

Theorem 2.16 *For any iteratable endofunctor H, the monad T is the free completely iterative monad on H. More precisely, for all completely iterative monads S and ideal transformations $\lambda : H \longrightarrow S$ there exists a unique idealized-monad morphism $\overline{\lambda} : T \longrightarrow S$ such that $\overline{\lambda} \cdot \tau^* = \lambda$:*



Remark 2.17

(i) Since the inclusion of the ideal $\sigma : S' \rightarrowtail S$ is a monomorphism, the last condition is equivalent to stating that



commutes, where $\overline{\lambda}' : HT \longrightarrow S'$ is the restriction of $\overline{\lambda}$ to the ideal of T.

(ii) Categorically, the statement of the theorem says that every iteratable functor H in $[\mathcal{A}, \mathcal{A}]$ has a universal arrow w. r. t. the forgetful functor

$$U : \mathbf{CIM}(\mathcal{A}) \longrightarrow [\mathcal{A}, \mathcal{A}]$$

of the category $\mathbf{CIM}(\mathcal{A})$ of all completely iterative monads and idealized-monad morphisms. Beware! The functor U assigns to every completely

iterative monad S its ideal S', not the underlying functor S. This choice of U corresponds to the requirement that $\lambda : H \longrightarrow S$ be an ideal transformation.

The above result states that any iteratable endofunctor admits a free completely iterative monad. However, the obvious question whether these are the only endofunctors with this property remains unanswered in [AAMV]. We will present this answer in the next section.

3 Iteratability is neccessary

We have seen above that any iteratable endofunctor admits a free completely iterative monad. We shall prove in this section that endofunctors that admit a free completely iterative monad are precisely the iteratable ones.

Throughout this section we shall denote by $\mathcal{A}$ a category with binary coproducts such that injections are monomorphic.

Theorem 3.1 *Every endofunctor generating a free completely iterative monad is iteratable.*

Remark 3.2 More detailed, suppose that H is an endofuntor on $\mathcal{A}$ and

$$\sigma^* : H \longrightarrow S$$

is a free completely iterative monad on H (where σ^* is an ideal transformation), then H is iteratable and for all objects X of $\mathcal{A}$, SX is a final coalgebra of $H(\_) + X$.

Before we proceed with the proof of this theorem, let us prove two auxilliary results. First we establish that for any natural transformation $H \longrightarrow S$, where H is any endofunctor and S any monad on $\mathcal{A}$, one can easily obtain an ideal monad $\widetilde{S}$ and an ideal transformation $H \longrightarrow \widetilde{S}$ as follows.

Definition 3.3 Let (S, η, μ) be a monad on $\mathcal{A}$ and let

$$\sigma^* : H \longrightarrow S$$

be a natural transformation from an endofuntor H on $\mathcal{A}$. Define $(\widetilde{S}, \widetilde{\eta}, \widetilde{\mu})$ as follows:

(i) $\widetilde{S} = HS + Id$

(ii) $\widetilde{\eta} \equiv \mathsf{inr} : Id \longrightarrow HS + Id$

(iii) $\widetilde{\mu} \equiv \widetilde{S}^2 = (HS + Id)^2 = HS(HS + Id) + HS + Id$

$$\downarrow{\scriptstyle HS(\sigma^* S+Id)+HS+Id}$$

$$HS(S^2 + Id) + HS + Id$$

$$\downarrow{\scriptstyle HS[\mu,\eta]+HS+Id}$$

$$HS^2 + HS + Id$$

$$\downarrow{\scriptstyle [H\mu,\mathsf{inl}]+Id}$$

$$HS + Id = \widetilde{S}$$

Lemma 3.4 *The triple $(\widetilde{S}, \widetilde{\eta}, \widetilde{\mu})$ is an ideal monad.*

Proof. Once we have established that $\widetilde{S}$ is a monad, it is obvious that it is ideal: Note that for $\widetilde{S}' = HS$ we have

$$\widetilde{\mu}' \equiv \widetilde{S}'\widetilde{S} = HS(HS + Id) \xrightarrow{HS(\sigma^* S+Id)} HS(S^2 + Id) \xrightarrow{HS[\mu,\eta]} HS^2 \xrightarrow{H\mu} HS = \widetilde{S}'.$$

Hence, it is sufficient to show that $\widetilde{\eta}$ and $\widetilde{\mu}$ satisfy the three axioms of a monad.

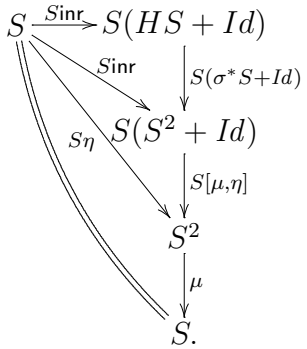
(i) $\widetilde{\mu} \cdot \widetilde{\eta}_{\widetilde{S}} = 1_{\widetilde{S}}$: This is obvious since

$$HS + Id \xrightarrow{\mathsf{inr}} HS(HS + Id) + HS + Id \xrightarrow{\widetilde{\mu}} HS + Id \equiv 1_{HS+Id}.$$

(ii) $\widetilde{\mu} \cdot \widetilde{S}\widetilde{\eta} = 1_{\widetilde{S}}$: Observe that

$$\widetilde{S}\widetilde{\eta} \equiv HS + Id \xrightarrow{HS\mathsf{inr}+\mathsf{inr}} HS(HS + Id) + HS + Id.$$

We compose this with $\widetilde{\mu}$ and consider the components of the coproduct $HS + Id$ separately. On the right-hand component we obviously obtain $\mathsf{inr} : Id \longrightarrow HS + Id$. For the left-hand one we drop H and consider the resulting commutative diagram



(iii) $\widetilde{\mu} \cdot \widetilde{S}\widetilde{\mu} = \widetilde{\mu} \cdot \widetilde{\mu}\widetilde{S}$: This is a straightforward and not particularly enlightening chase through rather huge diagrams. Since it only involves naturality and

the equation $\mu \cdot S\mu = \mu \cdot \mu S$, we leave this as an easy exercise for the Reader.

□

Lemma 3.5 *If S in Definition 3.3 above is a completely iterative monad and $\sigma^* : H \longrightarrow S$ is an ideal transformation, then $\widetilde{S}$ is completely iterative, too.*

Proof. We have to show that for each guarded equation morphism $e : X \longrightarrow \widetilde{S}(X+Y)$ with a factorization

$$\begin{array}{ccc} X & \xrightarrow{e} & HS(X+Y)+X+Y \\ & \searrow^{f} & \uparrow{\scriptstyle [\mathsf{inl},\mathsf{inr}]} \\ & & HS(X+Y)+Y \end{array}$$

we have a unique solution $e^\dagger : X \longrightarrow \widetilde{S}Y$. We define a guarded equation arrow $\overline{e} : X \longrightarrow S(X+Y)$ as follows

$$\overline{e} \equiv X \xrightarrow{f} HS(X+Y)+Y \xrightarrow{\sigma^* S+\eta} S^2(X+Y)+SY \xrightarrow{[\mu, S\mathsf{inr}]} S(X+Y).$$

In order to see that $\overline{e}$ is indeed guarded, use that S is an idealized monad and that σ^* is an ideal transformation.

We solve $\overline{e}$ to obtain a unique arrow $\overline{e}^\dagger : X \longrightarrow SY$ such that the outer shape of the following diagram

(2)

$$\begin{array}{ccccc} X & & \xrightarrow{\overline{e}^\dagger} & & SY \\ {\scriptstyle f}\downarrow & & & \nearrow{\scriptstyle [\mu,\eta]} & \uparrow{\scriptstyle \mu_Y} \\ & (II) & HSY+Y & \xrightarrow{\sigma^* S+Y} & S^2Y+Y \\ & & {\scriptstyle H\mu+Y}\uparrow & & \uparrow{\scriptstyle S\mu+Y} \\ HS(X+Y)+Y & \xrightarrow[HS[\overline{e}^\dagger,\eta_Y]+Y]{} & HS^2+Y & \xrightarrow[\sigma^* S^2+Y]{} & S^3Y+Y \\ {\scriptstyle \sigma^* S+\eta}\downarrow & & (I) & & \downarrow{\scriptstyle S^3Y+\eta} \\ S^2(X+Y)+SY & & \xrightarrow{S^2[\overline{e}^\dagger,\eta_Y]+SY} & & S^3Y+SY \\ {\scriptstyle [\mu,S\mathsf{inr}]}\downarrow & & & \searrow{\scriptstyle [\mu S, S\eta]} & \\ S(X+Y) & & \xrightarrow[S[\overline{e}^\dagger,\eta_Y]]{} & & S^2Y \end{array}$$

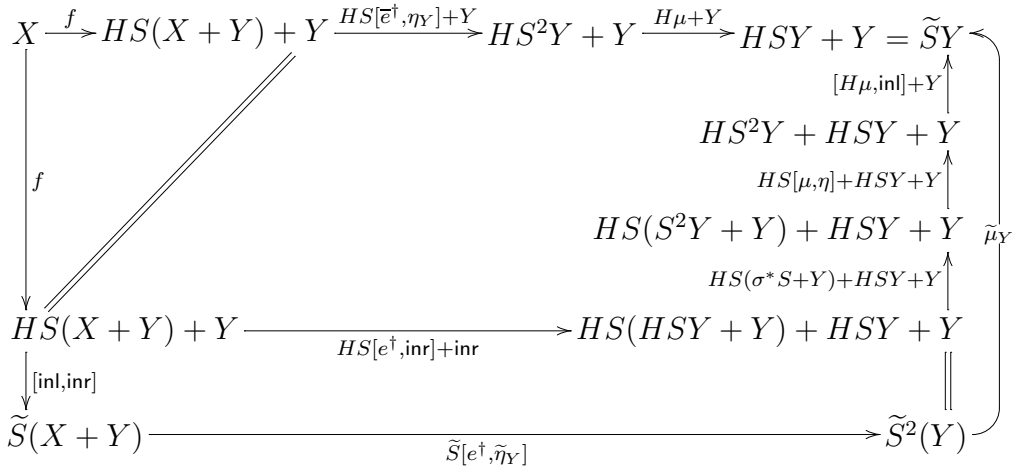
commutes. To see that square (I) commutes consider the components of the coproduct $HS(X+Y)+Y$ separately. The left-hand component commutes by naturality of σ^*, whereas the right-hand one obviously does. Hence, region (II) commutes since all other parts of the above Diagram (2) clearly do.

We define

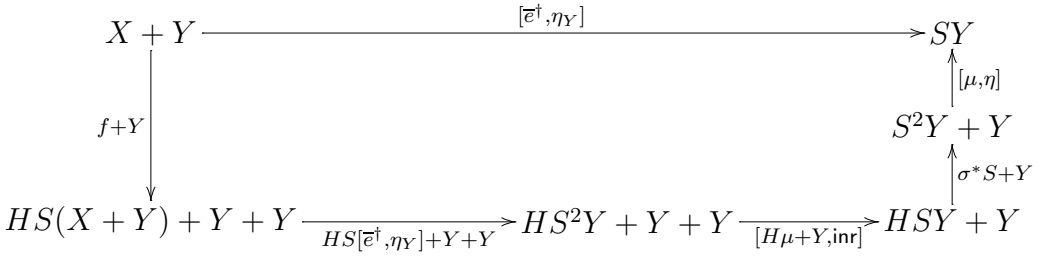
$$e^\dagger \equiv X \xrightarrow{f} HS(X+Y)+Y \xrightarrow{HS[\overline{e}^\dagger,\eta_Y]+Y} HS^2Y+Y \xrightarrow{H\mu+Y} HSY+Y = \widetilde{S}Y,$$

and check that this yields a solution for e. Indeed, consider the following

diagram:



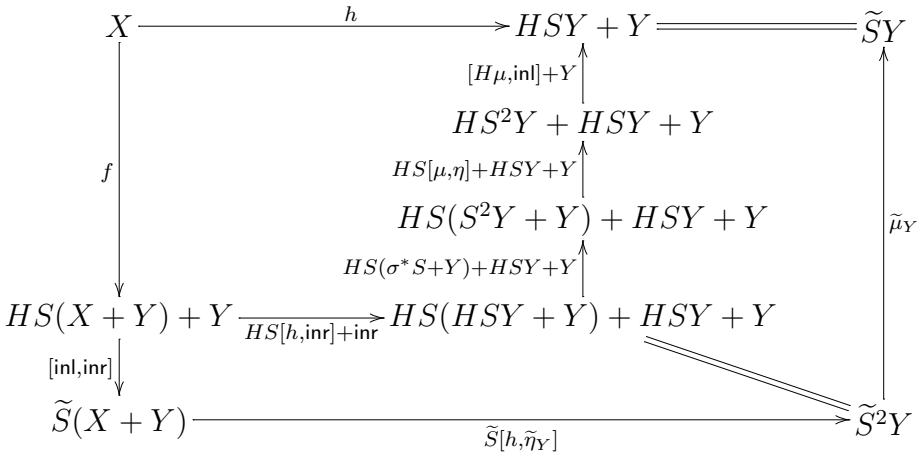
It obviously commutes, except perhaps the upper middle part. We consider its components separately. The right-hand component is the identity on Y. For the left-hand one notice that the last arrow is $H\mu$ on both paths. We show that the rest is already commutative, in fact, even if we drop HS. That is, we consider the resulting diagram:



This is obviously commutative. Indeed, the right-hand component is η_Y and the left-hand one is region (II) of diagram (2). This concludes the proof of the existence of a solution for e.

As for the unicity of solutions, consider any $h : X \longrightarrow \widetilde{S}Y$ such that the following diagram

(3)



commutes.

Below we will show that

$$X \xrightarrow{h} HSY + Y \xrightarrow{\sigma^* S + Y} S^2Y + Y \xrightarrow{[\mu,\eta]} SY \tag{4}$$

solves $\overline{e}$. But then it is not difficult to show that $e^\dagger = h$. In fact, we start with the definition of the solution $e^\dagger$

$$e^\dagger = (H\mu_Y + Y) \cdot (HS[\overline{e}^\dagger, \eta_Y] + Y) \cdot f,$$

then substitute (4) for $\overline{e}^\dagger$ to obtain

$$(H\mu_Y + Y) \cdot (HS\,[[\mu_Y, \eta_Y] \cdot (\sigma^*_{SY} + Y) \cdot h, \eta_Y] + Y) \cdot f, \tag{5}$$

and finally, we use the equation

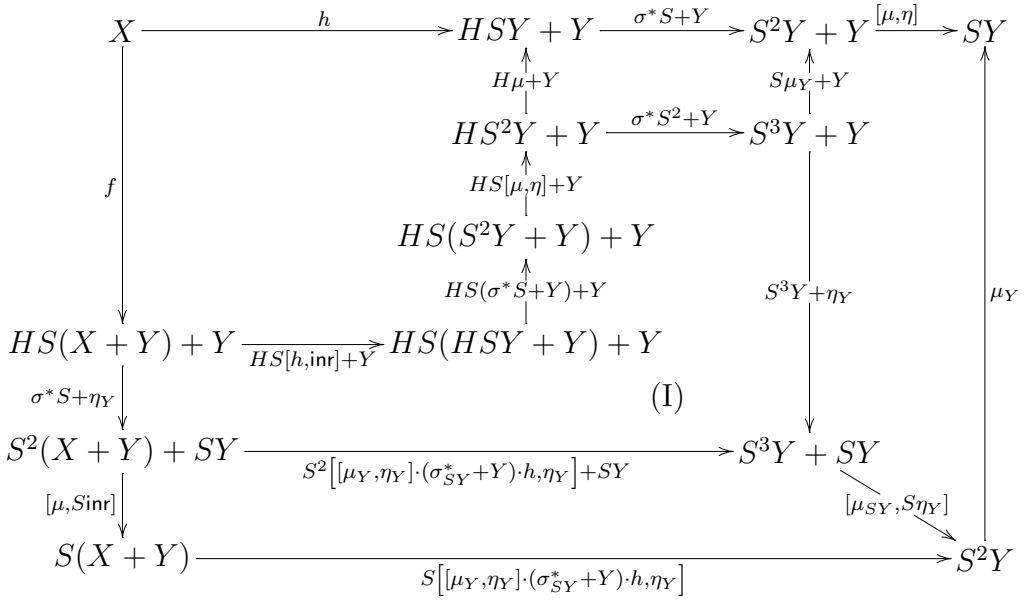
$$\eta_Y = [\mu_Y, \eta_Y] \cdot (\sigma^*_{SY} + Y) \cdot \mathsf{inr}. \tag{6}$$

in order to see that (5) is the same as

$$(H\mu_Y + Y) \cdot (HS[\mu_Y, \eta_Y] + Y) \cdot (HS(\sigma^*_{SY} + Y) + Y) \cdot (HS[h, \mathsf{inr}] + Y) \cdot f,$$

which according to the upper left-hand part of Diagram (3) is just h.

Let us complete our proof by showing that the arrow (4) solves $\overline{e}$. In fact, the following diagram



commutes. The upper left-hand square is just the upper left-hand square of Diagram (3). For the inner part (I), consider the components of the coproduct $HS(X + Y) + Y$ separately. The right-hand components obviously commute, for the left-hand ones use naturality of σ^* and Equation (6). All other parts clearly commute. □

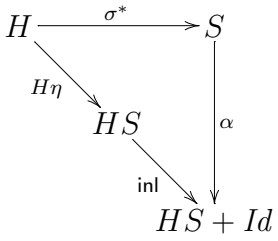
Proof of Theorem 3.1. Suppose that (S, η, μ) is a free completely iterative

monad on H, i. e., there exists a universal ideal transformation

$$\sigma^* : H \longrightarrow S.$$

By Lemma 3.4, $\widetilde{S} = HS + Id$ is an ideal monad, and by Lemma 3.5 it is completely iterative. Then by the universal property we have a unique idealized-monad morphism $\alpha : S \longrightarrow HS + Id$ such that the following diagram

(7)



commutes.

Note that for all objects Y of $\mathcal{A}$ the arrows

$$\alpha_Y : SY \longrightarrow HSY + Y$$

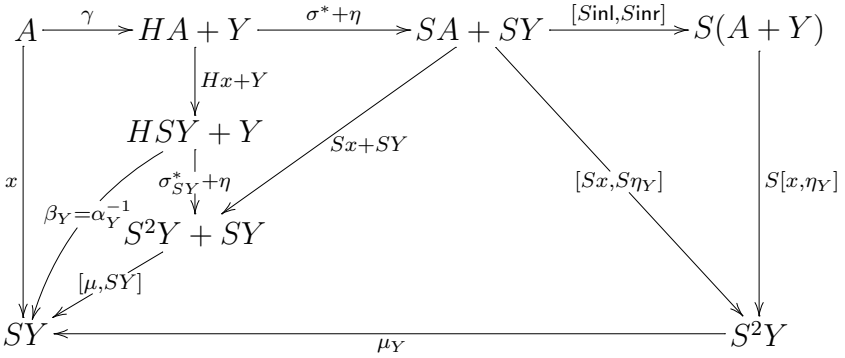
define a coalgebra structure for $H(\_) + Y$ on SY. We shall establish below that α is an isomorphism with an inverse given by the natural transformation

$$\beta \equiv HS + Id \xrightarrow{\sigma^* S + Y} S^2 + Id \xrightarrow{[\mu,\eta]} S.$$

In order to establish that (SY, α_Y) is a final coalgebra suppose that $\gamma : A \longrightarrow HA + Y$ is any coalgebra of $H(\_) + Y$. Then

$$\overline{\gamma} \equiv A \xrightarrow{\gamma} HA + Y \xrightarrow{\sigma^* + \eta} SA + SY \xrightarrow{[S\mathsf{inl}, S\mathsf{inr}]} S(A + Y)$$

is a guarded equation arrow (since σ^* is ideal, i. e., it factors through $\sigma : S' \longrightarrow S$) whose solution $\overline{\gamma}^\dagger : A \longrightarrow SY$ yields the desired unique homomorphism of coalgebras. Indeed, consider the following diagram:

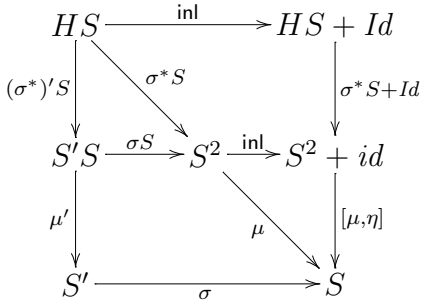


Suppose we put $\overline{\gamma}^\dagger$ in place of x in the diagram. Then the outer square commutes, and we conclude that the upper left-hand part commutes, since all other parts obviously do. This shows that $\overline{\gamma}^\dagger$ is a coalgebra homomorphism.

Conversely, put any coalgebra homomorphism $h : (A, \gamma) \longrightarrow (SY, \alpha_Y)$ in place of x. Then the upper left-hand part commutes, and therefore the whole diagram does. But then $h = \overline{\gamma}^\dagger$, by the uniqueness of solutions. This concludes the proof.

Finally, we show that β is the inverse of α.

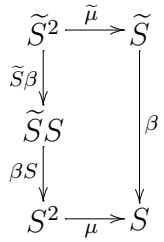
(i) $\beta \cdot \alpha = 1_S$: We will first show that $\beta : HS + Id \longrightarrow S$ is an idealized-monad morphism. In fact, once we know it is a monad morphism, it is easily established that it is ideal. To see this, consider the following commutative diagram:



Let us show that β is a monad homomorphism. We clearly have

$$\begin{aligned}\beta \cdot \widetilde{\eta} &= [\mu, \eta] \cdot (\sigma S + Id) \cdot \mathsf{inr} \\ &= [\mu, \eta] \cdot \mathsf{inr} \\ &= \eta.\end{aligned}$$

Hence, it suffices to prove that the following square



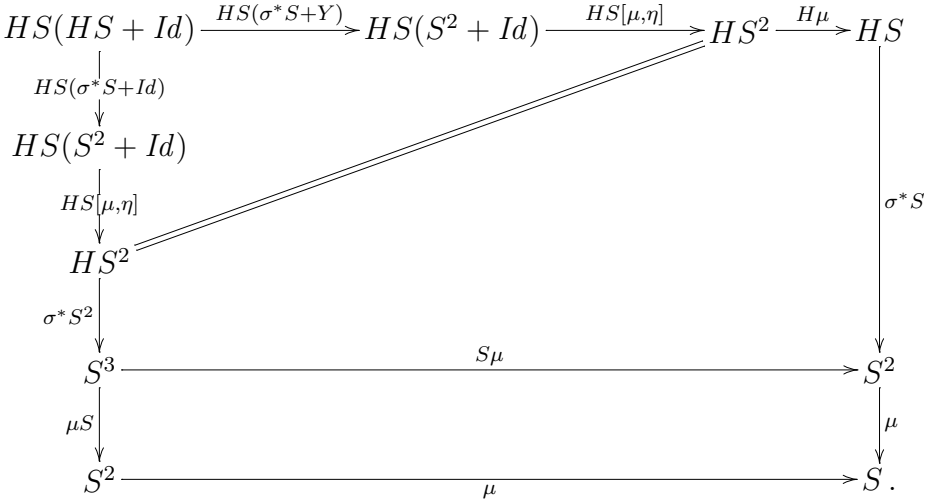
is commutative. We apply the definition of $(\widetilde{S}, \widetilde{\eta}, \widetilde{\mu})$ and consider the components of the coproduct

$$\widetilde{S}^2 = (HS + Id)^2 = HS(HS + Id) + HS + Id$$

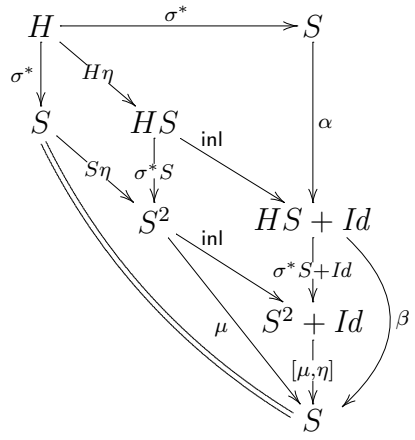
separately. For the right-hand component $HS + Id$ we obtain

$$\begin{aligned}\beta \cdot \widetilde{\mu} \cdot \mathsf{inr} &= \beta && (\mathsf{inr} = \widetilde{\eta}\widetilde{S}) \\ &= \mu \cdot \eta S \cdot \beta && (\mu \cdot \eta S = 1_S) \\ &= \mu \cdot \beta S \cdot \widetilde{\eta} S \cdot \beta && (\text{since } \beta \cdot \widetilde{\eta} = \eta) \\ &= \mu \cdot \beta S \cdot \widetilde{S}\beta \cdot \mathsf{inr} && (\text{naturality of } \widetilde{\eta}).\end{aligned}$$

For the left-hand component we obtain the following commutative diagram



Now $\beta \cdot \alpha$ is an idealized-monad morphism such that $\beta \cdot \alpha \cdot \sigma^* = \sigma^*$. In fact, the following diagram



commutes. Therefore, by the freeness of S on H, we have $\beta \cdot \alpha = 1_S$, as desired.

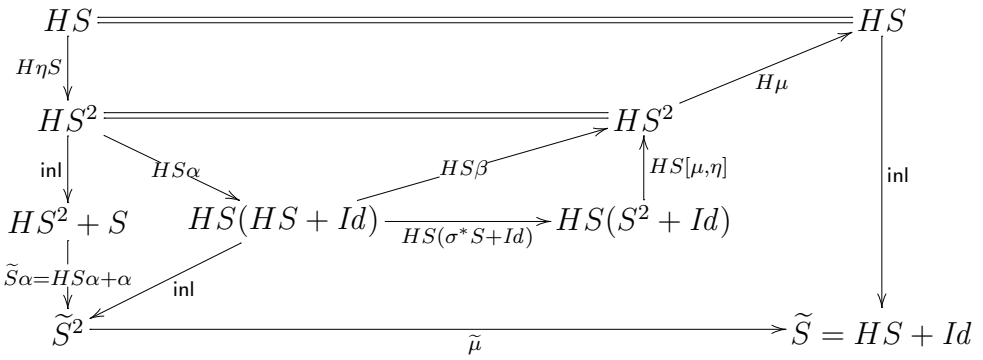
(ii) $\alpha \cdot \beta = 1_{HS+Id}$: We check this on the components of $HS + Id$. For the right-hand component we obtain

$$\begin{aligned} \alpha \cdot \beta \cdot \mathsf{inr} &= \alpha \cdot [\mu, \eta] \cdot (\sigma^* S + Id) \cdot \mathsf{inr} && \text{(definition of } \beta) \\ &= \alpha \cdot \eta \\ &= \mathsf{inr} && (\alpha \text{ is a monad morphism}), \end{aligned}$$

For the left-hand component we have

$$\begin{aligned}
\alpha \cdot \beta \cdot \mathsf{inl} &= \alpha \cdot [\mu, \eta] \cdot (\sigma^* S + Id) \cdot \mathsf{inl} && \text{(definition of } \beta) \\
&= \alpha \cdot \mu \cdot \sigma^* S \\
&= \widetilde{\mu} \cdot \widetilde{S}\alpha \cdot \alpha S \cdot \sigma^* S && (\alpha \text{ is a monad morphism}) \\
&= \widetilde{\mu} \cdot \widetilde{S}\alpha \cdot \mathsf{inl} \cdot H\eta S && (\alpha \cdot \sigma^* = \mathsf{inl} \cdot H\eta, \text{see (7)}).
\end{aligned}$$

We analyze the last expression further and obtain the following commutative diagram:



Note that the inner triangle commutes since $\beta \cdot \alpha = 1_S$, and the other parts obviously commute. Thus we have shown that

$$\alpha \cdot \beta = [\mathsf{inl}, \mathsf{inr}] = 1_{HS+Id}$$

as required.

□

Acknowledgments

I would like to thank Jiří Adámek for many useful suggestions that helped me improving this current text.

References

[AAV] P. Aczel, J. Adámek and J. Velebil, A Coalgebraic View of Infinite Trees and Iteration, *Electronic Notes in Theoretical Computer Science*, 44.1 (2001).

[AAMV] P. Aczel, J. Adámek, S. Milius and J. Velebil, Infinite Trees and Completely Iterative Theories: A Coalgebraic View, accepted for publication in *Theoretical Computer Science* .

[AK] J. Adamek and V. Koubek, On the greatest fixed point of a set functor, *Theoretical Computer Science*, 150 (1995), 57–75.

[ADJ] J. A. Goguen, S. W. Thatcher, E. G. Wagner and J. B. Wright, Initial Algebra Semantics and Continuous Algebras, *Journal of the ACM*, 24 (1977), 68–95.

[AMV] J. Adámek, S. Milius and J. Velebil, Free Iterative Theories: A Coalgebraic View, accepted for publication in *Math. Struct. in Comp. Science*.

[AP] J. Adámek and H. E. Porst, From Varieties of Algebras to Covarieties of Coalgebras, *Electronic Notes in Theoretical Computer Science*, 44.1 (2001).

[ARu] P. America and J. Rutten, Solving Reflexive Domain Equations in a Category of Complete Metric Spaces, *J. Comp. System Sci.*, 39 (1989), 343–375.

[CLW] A. Carboni, S. Lack and R. F. C. Wolters, Introduction to Extensive and Distributive Categories, *J. Pure Appl. Algebra*, 84 (1993), 145–158.

[E] C. C. Elgot, Monadic Computation and Iterative Algebraic Theories, in: *Logic Colloquium '73* (eds: H. E. Rose and J. C. Shepherdson), North-Holland Publishers, Amsterdam, 1975.

[EBT] C. C. Elgot, S. L. Bloom, R. Tindell, On the Algebraic Structure of Rooted Trees, *J. Comp. Syst. Sciences*, 16 (1978), 361–399.

[L] J. Lambek, A Fixpoint Theorem for Complete Categories, *Math. Z.*, 103 (1968), 151–161.

[Lin] F. E. J. Linton, Some aspects of equational categories, *Proceedings of the Conference on Categorical Algebra*, Springer-Verlag, 1966, 84–94.

[M] L. Moss, Parametric Corecursion, *Theoretical Computer Science*, 260 (2001), no. 1–2, 139–163.

[T] J. Tiuryn, Unique Fixed Points vs. Least Fixed Points, *Theoretical Computer Science*, 12 (1980), 229–254.

THE CATEGORY THEORETIC SOLUTION OF RECURSIVE PROGRAM SCHEMES

STEFAN MILIUS AND LAWRENCE S. MOSS

Abstract. This paper provides a general account of the notion of recursive program schemes, studying both uninterpreted and interpreted solutions. It can be regarded as the category-theoretic version of the classical area of algebraic semantics. The overall assumptions needed are small indeed: working only in categories with "enough final coalgebras" we show how to formulate, solve, and study recursive program schemes. Our general theory is algebraic and so avoids using ordered, or metric structures. Our work generalizes the previous approaches which do use this extra structure by isolating the key concepts needed to study substitution in infinite trees, including second-order substitution. As special cases of our interpreted solutions we obtain the usual denotational semantics using complete partial orders, and the one using complete metric spaces. Our theory also encompasses implicitly defined objects which are not usually taken to be related to recursive program schemes. For example, the classical Cantor two-thirds set falls out as an interpreted solution (in our sense) of a recursive program scheme.

Contents

Date: 16 Dezember 2005.
Key words and phrases. recursive program scheme, Elgot algebra, coalgebra, completely iterative monad, algebraic trees, second-order substitution .

1. Introduction

The theory of *recursive program schemes* is a topic at the heart of semantics. One takes a system of equations such as

$$\begin{array}{rcl} \varphi(x) & \approx & F(x, \varphi(Gx)) \\ \psi(x) & \approx & F(\varphi(Gx), GGx) \end{array} \tag{1.1}$$

where F and G are *given* functions and where φ and ψ are defined in terms of them by (1.1). The problems are: to give some sort of semantics to schemes, and to say what it means to *solve* a scheme. Actually, we should distinguish between *interpreted* schemes, and *uninterpreted* schemes.

An interpreted scheme is one which comes with an algebra with operations for all the given operation symbols. Here is the standard example in the subject. Let Σ be the signature of given operation symbols with a constant one, one unary symbol pred, a binary symbol $*$ and a ternary one ifzero. The interpretation we have in mind is the natural numbers where $\mathsf{ifzero}_{\mathbb{N}}(k, n, m)$ returns m if k is 0 and n otherwise, and all other operations are obvious. The signature Φ of the recursively defined operations consists just of one unary symbol f. Consider the recursive program scheme

$$f(n) \approx \mathsf{ifzero}(n, \mathsf{one}, f(\mathsf{pred}(n)) * n)) \,. \tag{1.2}$$

Then (1.2) is a recursive program scheme defining the factorial function.

This paper presents a generalization of the classical theory based on *Elgot algebras* and *final coalgebras*. The point in a nutshell is that knowing that the infinite trees are the final coalgebra of a functor on sets leads to a purely algebraic account of first-order substitution and (co-)recursion, as shown in [AAMV, Mo_1]. One does not need to assume any metric or order to study infinite trees: the finality principle is sufficient. In this paper we show that corecursion allows us to give an uninterpreted semantics to a scheme; i. e., we show how to solve a scheme in final coalgebras.

For our interpreted semantics we work with Elgot algebras, a simple and fundamental notion introduced in [AMV_3]. We show how to give an interpreted solution to recursive program schemes in arbitrary Elgot algebras. We believe that our results in this area generalize and extend the previous work on this topic. Furthermore, we claim that our abstract categorical approach allows a unified view of several well-known approaches to the semantics of implicit recursive function definitions. Our method for obtaining interpreted solutions easily specializes to the usual denotational semantics using complete partial orders. As a second application we show how to solve recursive program schemes in complete metric spaces. For example, there is a unique contracting $f : [0, 1] \longrightarrow [0, 1]$ such that

$$f(x) = \frac{1}{4}\left(x + f\left(\frac{1}{2}\sin x\right)\right). \tag{1.3}$$

Concerning sets of real numbers, recall that the Cantor set c is the unique non-empty closed $c \subseteq [0, 1]$ such that

$$c = \frac{1}{3}c \;\cup\; \left(\frac{2}{3} + \frac{1}{3}c\right), \tag{1.4}$$

where $\frac{1}{3}c$ denotes the set $\{\, \frac{1}{3}x \mid x \in c \,\}$ as usual.

Finally, our theory also encompasses examples of recursive program schemes and their solutions which cannot be treated with the classical theory; in this paper we present recursive definitions of operations satisfying equations like commutativity. Other examples are recursive program scheme solutions in non-wellfounded sets (solving $x = \{\, \{\, y \mid y \subseteq x \text{ finite} \,\} \,\}$). We will present the applications of our results to non-wellfounded sets in a future paper.

Our purpose in this paper is to isolate general principles which imply the existence and uniqueness results for uninterpreted solutions of systems such as (1.1) and interpreted schemes such as (1.2–1.4).

1.1. Several Semantics.

There are several semantics for recursive program schemes, and it is worth mentioning a bit about them, both to situate our work and also because we shall be interested in showing that one can recover different flavors of semantics from our more general approach.

1.1.1. *Operational Semantics.* This gives semantics to interpreted schemes only. Solutions are defined by rewriting. In our factorial example, the semantics of (1.2) would be as follows: to compute the solution $f^\dagger(n)$ for a natural number n start with the term $f(n)$, substitute it by its right-hand side in the scheme and evaluate this term as much as possible. If the evaluated right-hand side still contains f replace it again by its right-hand side and then evaluate the whole term, and so on. If after finitely many steps this process returns a natural number we have computed $f^\dagger(n)$; otherwise we declare $f^\dagger(n)$ to be undefined. (Of course there are important and subtle points to be considered here pertaining to issues like call by name and call by

value interpretations of function symbols, and also about the overall strategy of rewriting.) In the factorial example the described process always stops. But in general this may not be the case.

1.1.2. *Denotational Semantics.* Again this provides semantics for interpreted schemes only. The algebra that comes as the interpretation of the given functions is a complete partial order A with continuous operations for all given operation symbols. A system like (1.1) gives then rise to a continuous function R on a function space. In our factorial example from (1.2) one would consider the natural numbers as a flat cpo and R assigns to a continuous function f on that cpo the function

$$Rf(n) = \begin{cases} \bot & \text{if } n = \bot \\ 1 & \text{if } n = 0 \\ f(n-1) \cdot n & \text{else} \end{cases} \tag{1.5}$$

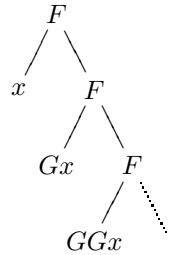
The semantics of the given scheme is then the least fixed point of R. More generally, denotational semantics provides a continuous operation φ_A on A for each newly defined operation symbol φ of a given recursive program scheme.

It is true but by no means obvious that the operational and denotational semantics agree in the appropriate sense. Thus there is a matter of semantic equivalence to be investigated. This was one of the primary starting points of the original theory of recursive program schemes, see [C, G, N]. In any case, there are two general themes that are raised by this matter of equivalence. First, one sees that the very definition of the denotational semantics requires order-theoretic methods. From our point of view, one must ask whether this is really necessary. One of the themes of work in coalgebra is that for some semantic problems, order-theoretic methods may be replaced by purely algebraic ones. This is a theme in some of the recent work in coalgebra that is relevant to our study. The reason is that coalgebra is often about semantics spaces for infinite behaviors of one type or another, and these are studied using universal properties (typically finality) instead of the extra structure coming from a cpo or complete metric space. The second general theme is what we might call *recovery of one semantics by another*. The first instance of this is the recovery of the operational semantics by the denotational semantics. One therefore feels that the denotational semantics is getting at something deeper. (At the same time, the very real subtle points of operational semantics are not easy to address with the methods of denotational semantics, so there is a loss as well.)

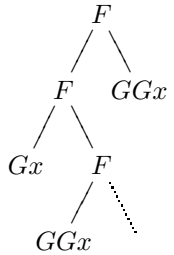
1.1.3. *Algebraic Semantics.* This considers *uninterpreted* recursive program schemes; i. e., where no interpretation is given. It is then an issue in this approach to make sure that one can recover the denotational and operational semantics inside the algebraic semantics. But first, one must say what a solution to (1.1) should be. Normally, one considers for a signature Σ and a set X of generators the set $T_\Sigma X$ of all finite and infinite Σ-trees over X, i. e., ordered and rooted trees labelled so that an inner node with n children, $n > 0$, is labelled by an n-ary operation symbol from Σ, and leaves are labelled by a constant symbol or a variable of X. Then one defines an appropriate notion of *second-order substitution* whereby Σ-trees may be substituted for operation symbols in Γ-trees for another signature Γ. In general, a recursive program scheme is given by two signatures Σ (of given operations) and Φ (of recursively defined operation), and a set of formal equations providing for each n-ary operation φ from Φ on the left-hand side a $(\Sigma + \Phi)$-tree over n syntactic variables on the right-hand side of the equation. A solution for such a recursive program scheme then assigns to each n-ary operation symbol φ from Φ a Σ-tree $\varphi^\dagger(x_1, \ldots, x_n)$ in n syntactic variables which is equal to the Σ-tree obtained by second-order substituting the solutions in the right-hand side of the formal equation defining φ.

As an example, consider the signature Σ with a binary operation symbol F and a unary symbol G. One might want to define new operations φ and ψ recursively as in (1.1) above. Notice that the system is *guarded*, or in *Greibach normal form*: the right-hand sides start with a symbol from Σ. One key opening result of algebraic semantics is that a unique solution exists for guarded systems among Σ-trees. For instance, for the above system (1.1) the solution consists of the Σ-trees

$\varphi^\dagger(x) =$



$\psi^\dagger(x) =$



(1.6)

This solution can in general be obtained as follows: the given scheme expands to a system of equations with recursion variables for each $(\Sigma + \Phi)$-tree t. This system gives to each recursion variable $\underline{t}$ where t is just a syntactic variable that variable itself, and otherwise the right-hand side of the system is given by replacing all appearances of symbols of Φ by their right-hand sides in the given scheme. For example, from (1.1) we obtain the equations

$$\begin{array}{rcl@{\qquad}rcl}
\underline{x} & \approx & x & \underline{\varphi(Gx)} & \approx & F(\underline{Gx}, \underline{\varphi(GGx)}) \\
\underline{\varphi(x)} & \approx & F(\underline{x}, \underline{\varphi(Gx)}) & \underline{F(x, \varphi(Gx))} & \approx & F(\underline{x}, \underline{F(Gx, \varphi(GGx))}) \\
\underline{\psi(x)} & \approx & F(\underline{\varphi(Gx)}, \underline{GGx}) & \underline{GGx} & \approx & G(\underline{Gx}) \\
\underline{\varphi(\psi(x))} & \approx & F(\underline{F(\varphi(Gx, GGx))}, \underline{\varphi(G(F(\varphi(Gx), GGx)))}) & & & \\
 & \vdots & & & &
\end{array} \tag{1.7}$$

and so on. Notice that each tree (here written as terms) on the right-hand side is a Σ-tree which is either just a syntactic variable or *flat*; i. e., one operation symbol from Σ with recursion variables. The solution of φ is now just the tree unfolding of the recursion variable $\underline{\varphi(x)}$ and similarly for ψ.

At this point, we have explained *how* we solve recursive program schemes in the algebraic semantics. But much has been omitted. For example, we gave no general account of *why* any solution method works, or even, why the above trees solve the given scheme.

The standard approach views infinite trees as either the completion of the finite trees, considered as a metric space, or else as the ideal completion of the set of finite trees. Second-order substitution is defined and studied using one of those methods, and using this one can say what it means to solve a recursive program scheme. We shall show in this paper that all of this can be done more generally and conceptually much nicer by considering Σ-trees as final coalgebras. More precisely, the algebra $T_\Sigma X$ of all Σ-trees over X is the final coalgebra for the functor $H_\Sigma(\_) + X$, where H_Σ is the polynomial endofunctors on sets associated to the signature Σ. It is the universal property of those final coalgebras, for any set X, which allows us to give a semantics to recursive program schemes.

1.1.4. *Category Theoretic Semantics.* As the title of our paper suggests, our goal is to propose a category-theoretic semantics of program schemes. The idea is to be even deeper than the algebraic semantics, and to therefore obtain results that are more general. Our theory is based on notions from category theory (monads, Eilenberg-Moore algebras) and coalgebra (finality, solution theorems, Elgot algebras). The overall assumptions are weak: there must be finite coproducts, and all functors we deal with must have "enough final coalgebras". More precisely, we work in a category $\mathcal{A}$ with finite coproducts and with functors $H : \mathcal{A} \longrightarrow \mathcal{A}$ such that for all objects X a final coalgebra TX of $H(\_) + X$ exists. We shall introduce and study recursive program schemes in this setting. In particular, we are able to prove the existence and uniqueness of interpreted and uninterpreted solutions to schemes. The price we pay for working in such a general setting is that our theory takes somewhat more effort to build. But this is not excessive, and perhaps our categorical proofs reveal more conceptual clarity than the classical ones.

Further, the issue of semantic recovery is quite interesting in our study. As we mentioned, we can recover the key aspects of the algebraic semantics in our approach. It follows that we can recover the other semantics as well. We shall interpret schemes in Elgot algebras. One of the key examples is the algebra $T_\Sigma X$ of all Σ-trees over X; and indeed, it appears that the fact that this is a free Elgot algebra on X implies all of the structural properties that are of interest when studying recursion. But in addition, there are many other interesting examples of Elgot algebras. As it happens, continuous algebras are Elgot algebras. We shall show that our general results on solving recursive program schemes in Elgot algebras directly specializes to *least fixed-point recursion* in continuous algebras. This yields the usual application of recursive program scheme solutions for the semantics of functional programs such as (1.2). Furthermore, algebras on complete metric spaces with contracting operations are Elgot algebras, and so our results specialize to the *unique fixed-point recursion* in completely metrized algebras provided by Banach's fixed-point theorem. This yields applications such as the ones in (1.3) and (1.4) above.

Related Work. The classical theory of recursive program schemes is compactly presented by Guessarian [G]. There one finds results on uninterpreted solutions of program schemes and interpreted ones in continuous algebras.

The first categorical accounts of infinite trees as monads of final coalgebras appear independently at almost at the same time in work of the second author [Mo_1], in Ghani et al. [GLMP_1, GLMP_2], and in Aczel et al. [AAV]. Furthermore, in [Mo_1] and in [AAV, AAMV] it is shown how solutions of formal recursive equations can be studied with coalgebraic methods. And in [AAMV] and [Mi_1] of the first author the

monad of final coalgebras is characterized as the free completely iterative monad, generalizing and extending results of Elgot et al. [E, EBT]. Hence, from [AAMV, Mi_1] it also follows how to generalize *second-order substitution* of infinite trees (see Courcelle [C]) to final coalgebras. The types of recursive equations studied in [Mo_1, AAMV] did not go as far as program schemes. It is thus an important test problem for coalgebra to see if work on solving systems of equations can extend to (un)interpreted recursive program schemes. We are pleased that this paper reports a success in this matter.

Ghani et al. [GLM] obtained a general solution theorem with the aim of providing a categorical treatment of uninterpreted program scheme solutions. Part of our proof for the solution theorem for uninterpreted schemes is inspired by their proof of the same fact. However, the notion of recursive program scheme in [GLM] is different from ours, and more importantly, our presentation of recursive program scheme solutions as fixed points with respect to to (generalized) second-order substitution as presented in [AAMV] is new here.

Complete metric spaces as a basis for the semantics of recursive program schemes have been studied for example by Arnold and Nivat [AN]. Bloom [Bl] studied interpreted solutions of recursive program schemes in so-called contraction theories. The semantics of recursively defined data types as fixed points of functors on the category of complete metric spaces has been investigated by Adámek and Reitermann [ARe] and by America and Rutten [ARu]. We build on this with our treatment of self-similar objects. These have also recently been studied in a categorical framework by Leinster, see [L_1, L_2, L_3]. The example in this paper uses standard results on complete metric spaces, see, e.g. [B]. We are not aware of any work on recursive program schemes that even mentions connections to examples like self-similar sets in mathematics, let alone develops the connection.

Finally, some of the results of this paper have appeared in our extended abstract [MM]. However, most of the technical details and all of the proofs were omitted. This full version explains our theory in much more detail and provides full proofs of all results.

Contents. The paper is structured as follows: Section 2 contains a brief summary of the definitions concerning monads. It also states the overall assumptions that we make in the rest of the paper. Section 3 presents the notions of *completely iterative algebra* and *Elgot algebra*, following [AMV_3]. Except for the Section 3.3, none of the results in this section is new here. But the paper as a whole would not make much sense to someone unfamiliar with completely iterative algebras and Elgot algebras. So we have included sketches of proofs in this section. The completely iterative monads of Section 4 are also not original here. But the properties of them developed in Section 5 are new. These are needed for the work on uninterpreted schemes (Section 6) and then interpreted schemes (Section 7). These are the heart of the paper.

2. Background: Iteratable Functors and Monads

This section contains most of the background which we need and also connects that background with recursive program schemes. Before reviewing monads in Section 2.2 we should mention the main base categories of interest in this paper, and our overall assumptions on those categories and endofunctors on them.

2.1. **Iteratable Functors.** Throughout this paper we assume that a category $\mathcal{A}$ with finite coproducts (having monomorphic injections) is given. In addition, all endofunctors H on $\mathcal{A}$ we consider are assumed to be *iteratable* [sic]: for each object X, the functor $H(\_) + X$ has a final coalgebra. We denote for an iteratable endofunctor H on $\mathcal{A}$ by

$$\mathcal{T}(H)X$$

the final coalgebra of $H(\_) + X$. Whenever confusion is unlikely we will drop the parenthetical (H) and simply write T for $\mathcal{T}(H)$. By the Lambek Lemma [L], the structure morphism of the final coalgebra is an isomorphism, and consequently, TX is a coproduct of HTX and X with injections

$$\begin{array}{lll} \eta^H_X & : X \longrightarrow TX & \text{“injection of variables”} \\ \tau^H_X & : HTX \longrightarrow TX & \text{“}TX\text{ is an }H\text{-algebra”} \end{array}$$

Again, the superscripts will be dropped if confusion is unlikely.

Having coproduct injections that are monomorphic is a mere technicality and could even be totally avoided, see [Mi_2], Sections 4 and 5. However, this assumption simplifies our presentation.

More serious is the assumption of iteratability. In one sense, this is a mild assumption: experience shows most endofunctors of interest are iteratable.

Example 2.1. We recall that ordinary signatures of function symbols Σ (as in universal algebra) give functors on Set. This is one of the central examples in this paper because it will allow us to recover the classical algebraic semantics from our category-theoretic one. A signature may be regarded as a functor

$\Sigma : \mathbb{N} \longrightarrow \mathsf{Set}$, where $\mathbb{N}$ is the discrete category with natural numbers as objects. For each n, write Σ_n for $\Sigma(n)$; this is the set of function symbols of arity n in Σ. There is no requirement that different numbers should give disjoint sets of function symbols of those arities. Given any signature Σ there is an associated polynomial endofunctor (henceforth called a *signature functor*)

$$H_\Sigma X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X^2 + \cdots \tag{2.1}$$

on Set. When we need to refer to elements of $H_\Sigma X$, we shall use the notation $(f, \vec{x})$ for a generic element of $H_\Sigma X$; n is understood in this notation, $f \in \Sigma_n$, and $\vec{x}$ is an n-tuple of elements of X. Also observe that on morphisms $k : X \longrightarrow Y$ the action of H_Σ is given by $H_\Sigma k(f, \vec{x}) = (f, \overrightarrow{kx})$.

Signature functors H_Σ of Set are iteratable. In fact, consider the set

$$T_\Sigma X$$

of finite and infinite Σ-labelled trees with variables from the set X. That is, ordered and rooted trees labelled so that a node with n children, $n > 0$, is labelled by an operation symbol from Σ_n, and leaves are labelled by constant symbols or variables (elements of $X + \Sigma_0$).

This set $T_\Sigma X$ is the carrier of a final coalgebra for $H_\Sigma(\_) + X$. The coalgebra structure is the inverse of tree tupling paired with the obvious injection $\eta_X : X \longrightarrow T_\Sigma X$ which regards each variable as a one-point tree.

So at this point we have seen signature functors on Set as an examples of iteratable functors. Unfortunately, we must admit that iteratability is not a very nice notion with respect to closure properties of functors—for example, iteratable functors need not be closed under coproducts or composition. The main examples of base categories $\mathcal{A}$ in this paper are Set, CPO and CMS. In these categories there are stronger yet much nicer conditions that ensure iteratability.

Examples 2.2. (i) Accessible endofunctors of Set. Let λ be a regular cardinal. An endofunctor H of the category Set of sets and functions is called λ-*accessible* it it preserves λ-filtered colimits. It is shown in [AP], Proposition 5.2, that λ-accessible functors are precisely those endofunctors where for each set X any element $x \in HX$ lies in the image of $Hm : HY \longrightarrow HX$ for some subset $Y \hookrightarrow X$ of cardinality less than λ. As usual, we call an endofunctor H *finitary* if it is ω-accessible and we call H *accessible* if it is λ-accessible for some regular cardinal λ. Any accessible endofunctor is iteratable, see [Ba]. In particular, signature functors are finitary, whence iteratable (as we already know).

(ii) CPO is the category of ω-complete partial orders, i. e., partially ordered sets with joins of all increasing ω-chains (but not necessarily with a least element $\bot$). Morphisms of CPO are the continuous functions (preserving joins of all ω-chains). Hence the morphisms are monotone (preserve the order). Notice that coproducts in CPO are disjoint unions with elements of different summands incompatible. CPO is understood to be enriched in the obvious way. For given cpo's X and Y, the hom-set $\mathsf{CPO}(X, Y)$ is a cpo in the pointwise order. An endofunctor H on CPO is *locally continuous* if it preserves the extra structure just noted, i. e., each derived function $\mathsf{CPO}(X, Y) \longrightarrow \mathsf{CPO}(HX, HY)$ is continuous. Observe that not all locally continuous functors need be iteratable. For a counterexample consider the endofunctor assigning to a cpo X the powerset of the set of order components of X. This is a locally continuous endofunctor but it does not have a final coalgebra. However, any accessible endofunctor H on CPO has a final coalgebra, see [Ba], and moreover, H is iteratable.

(iii) Finally, CMS is the category of complete metric spaces with distances measured in the interval $[0, 1]$ together with *non-expanding* maps $f : X \longrightarrow Y$; i. e., $d_Y(fx, fy) \leq d_X(x, y)$ for all $x, y \in X$. A stronger condition is that f be ε-*contracting*: for some $\varepsilon < 1$ such that $d_Y(fx, fy) \leq \varepsilon \cdot d_X(x, y)$ for $x, y \in X$. Again, CMS is understood to be enriched. For complete metric spaces (X, d_X) and (Y, d_Y) the hom-set $\mathsf{CMS}(X, Y)$ is a complete metric space with the metric given by

$$d_{X,Y}(f, g) = \sup_{x \in X} d_Y(f(x), g(x)) \,.$$

Recall that a functor H on CMS is called ε-*contracting* if there exists a constant $\varepsilon < 1$ such that each derived function $\mathsf{CMS}(X, Y) \longrightarrow \mathsf{CMS}(HX, HY)$ is an ε-contracting map; i. e.,

$$d_{HX,HY}(Hf, Hg) \leq \varepsilon \cdot d_{X,Y}(f, g)$$

for all non-expanding maps $f, g : X \longrightarrow Y$. Contracting functors on CMS are iteratable, see [ARe].

Construction 2.3. Let H be an endofunctor of Set. We recall here that a final coalgebra T for H can (if it exists) be constructed as a limit of an (op-)chain. Let us define by transfinite induction the following chain indexed by all ordinal numbers:

$$\begin{array}{lllllll}
\text{Initial step:} & T_0 & = & 1, & t_{1,0} & \equiv & H1 \xrightarrow{!} 1, \\
\text{Isolated step:} & T_{i+1} & = & HT_i, & t_{i+1,j+1} & \equiv & HT_i \xrightarrow{Ht_{i,j}} HT_j \\
\text{Limit step:} & T_j & = & \lim\limits_{i<j} T_i & \text{with limit cone } t_{j,i} : T_j \longrightarrow T_i, & & i<j,
\end{array}$$

where the connecting map $t_{j+1,j}$ is uniquely determined by the commutativity of the squares

$$\begin{array}{ccc}
T_{i+1} = HT_i & \xleftarrow{Ht_{j,i}} & HT_j = T_{j+1} \\
{\scriptstyle t_{i+1,i}}\downarrow & & \vdots\,{\scriptstyle t_{j+1,j}} \\
T_i & \xleftarrow[t_{j,i}]{} & T_j
\end{array} \qquad i<j\,.$$

This chain it said to *converge* if $t_{i+1,i}$ is an isomorphism for some ordinal number i.

It has been proved by Adámek and Koubek [AK] that an endofunctor H has a final coalgebra iff the above chain converges; moreover, if i is an ordinal number such that $t_{i+1,i}$ is invertible, then $t_{i+1,i}^{-1} : T_i \longrightarrow HT_i$ is a final coalgebra for H. For many set endofunctors one can give a bound for the number of steps it will take until the above final coalgebra chain T_i converges. The following result has been established by Worrell [W].

Theorem 2.4. *For a λ-accessible endofunctor H of Set the final coalgebra chain T_i converges after $\lambda \cdot 2$ steps and $(T_{\lambda\cdot 2}, t_{\lambda\cdot 2+1,\lambda\cdot 2}^{-1})$ is a final coalgebra for H.*

In particular, for a finitary endofunctor a final coalgebra is obtained after $\omega + \omega$ steps. For some functors one can further improve on this bound. For endofunctors preserving limits of countable op-chains the final coalgebra chain converges after countably many steps so that $(T_\omega, t_{\omega+1,\omega}^{-1})$ is a final coalgebra. For example, each signature functor H_Σ of Set preserves limits of op-chains.

Examples 2.5. We mention additional examples of iteratable endofunctors H with their final coalgebras TX on the categories of interest.

(i) A functor $H : \mathsf{Set} \longrightarrow \mathsf{Set}$ is finitary (i. e., it preserves filtered colimits) iff it is a quotient of some polynomial functor H_Σ, see [AT], III.4.3. The latter means that we have a natural transformation $\varepsilon : H_\Sigma \longrightarrow H$ with epimorphic components ε_X. These components are fully described by their kernel equivalence whose pairs can be presented in the form of so-called basic equations

$$\sigma(x_1, \ldots, x_n) = \rho(y_1, \ldots, y_m)$$

for $\sigma \in \Sigma_n$, $\rho \in \Sigma_m$ and $(\sigma, \vec{x}), (\rho, \vec{y}) \in H_\Sigma X$ for some set X including all x_i and y_j. Adámek and Milius [AM] have proved that the final coalgebra TX of $H(\_) + X$ is given by the quotient $T_\Sigma X/{\sim_X}$ where $\sim_X$ is the following congruence: for every Σ-tree t denote by $\partial_n t$ the finite tree obtained by cutting t at level n and labelling all leaves at that level by some symbol $\perp$ not from Σ. Then we have $s \sim_X t$ for two Σ-trees s and t iff for all $n < \omega$, $\partial_n s$ can be obtained from $\partial_n t$ by finitely many applications of the basic equations describing the kernel of ε_X. For example, the functor H which assigns to a set X the set $\{\, \{\, x, y \,\} \mid x, y \in X \,\}$ of unordered pairs of X is a quotient of $H_\Sigma X = X \times X$ expressing one binary operation b where ε_X is presented by commutativity of b; i. e., by the basic equation $b(x, y) = b(y, x)$. And TX is the coalgebra of all unordered binary trees with leaves labelled in the set X.

(ii) Consider the finite power set functor $\mathcal{P}_\mathrm{f} : \mathsf{Set} \longrightarrow \mathsf{Set}$. Under the Anti-Foundation Axiom (AFA), its final coalgebra is the set HF_1 of hereditarily finite sets; see [BM]. Analogously, the final coalgebra of $\mathcal{P}_\mathrm{f}(\_) + X$ is the set $HF_1(X)$ of hereditarily finite sets generated from the set X. Even without AFA, the final coalgebra of $\mathcal{P}_\mathrm{f}$ may be described as in Worrell [W]; it is the coalgebra formed by all strongly extensional trees; i. e., unordered trees so that for every node the subtrees defined by any two different children are not bisimilar. Analogously, the final coalgebra of $\mathcal{P}_\mathrm{f}(\_) + X$ is the coalgebra of all strongly extensional trees where some leaves have a label from the set X.

(iii) The (unbounded) power set functor $\mathcal{P} : \mathsf{Set} \longrightarrow \mathsf{Set}$ does not have a final coalgebra, whence it is not iteratable. However, moving to the category of classes and functions between them, the power set functor turns out to be iteratable, see e. g. [AMV$_3$]. Indeed, some of the machinery that comes from iteratable functors turns out to have a surprisingly set-theoretic interpretation when specialized to this setting; see [Mo$_3$].

(iv) In our applications, the key point is that certain Set endofunctors lift to (iteratable) endofunctors on CPO. And we need to know that those liftings are locally continuous. In fact, let H be an iteratable Set functor with a locally continuous lifting H' on CPO; i. e., a functor H' so that for the forgetful functor $U : \mathsf{CPO} \longrightarrow \mathsf{Set}$ we have a commutative square

$$\begin{array}{ccc} \mathsf{CPO} & \xrightarrow{H'} & \mathsf{CPO} \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathsf{Set} & \xrightarrow[H]{} & \mathsf{Set} \end{array}$$

Then H' is iteratable, and moreover, the final coalgebra functor $\mathcal{T}(H')$ is a lifting of $\mathcal{T}(H)$:

$$\begin{array}{ccc} \mathsf{CPO} & \xrightarrow{\mathcal{T}(H')} & \mathsf{CPO} \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathsf{Set} & \xrightarrow[\mathcal{T}(H)]{} & \mathsf{Set} \end{array} \tag{2.2}$$

To see this, first recall that for any set X the final coalgebra $\mathcal{T}(H)X$ is obtained from the final coalgebra chain T_i of $H(\_) + X$, see Construction 2.3. In fact, $\mathcal{T}(H)X$ is the coalgebra $(T_j, t_{j+1,j}^{-1})$ for the smallest ordinal number j for which $t_{j+1,j}$ is invertible. As CPO is a complete category we can define for any endofunctor H' of CPO a final coalgebra chain T_i' in precisely the same way as in 2.3. Since the forgetful functor U preserves limits, it follows that for a cpo X the final coalgebra chain of $H'(\_) + X$ has the T_i as underlying sets. However, in CPO the continuous map $t_{j+1,j}$ might not be invertible. But since the chain of underlying sets converges at index j we know that for all ordinal numbers k the connecting maps $t_{j+k,j} : T_{j+k} \longrightarrow T_j$ are monomorphisms of CPO. Moreover, all cpos T_{j+k} have (up to isomorphism) the same underlying set T_j and therefore the partial orders on the T_{j+k}, $k \geq 0$, form a decreasing chain of subsets of $T_j \times T_j$. This implies that the final coalgebra chain has to converge at some index $j + k$, $k \leq \mathrm{card}(T_j \times T_j)$. By standard arguments it follows that the cpo T_{j+k} is the final coalgebra of $H'(\_) + X$. Thus, we may choose $\mathcal{T}(H')X = T_j$ equipped with the cpo structure given by its subcpo T_{j+k}, whence the square (2.2) commutes as desired.

For example, every signature functor H_Σ has a locally continuous and iteratable lifting H'. This lift is the functor

$$H'X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X^2 + \cdots$$

on CPO. Here each Σ_n is a discrete ordered set, $+$ is the coproduct of CPO (a lift of the coproduct of Set) and $\times$ the usual product. It should be noted that even if X has a least element, $H'X$ almost never has one. Finally, $\mathcal{T}(H')X$ is the Σ-tree algebra $T_\Sigma X$ with the order induced by the order of the cpo X—we describe this order in more detail later in Example 7.13(i).

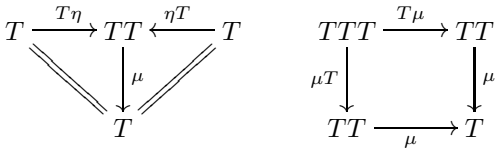
(v) Let $H : \mathsf{Set} \longrightarrow \mathsf{Set}$ have a contracting lifting H' on CMS; i. e.,

$$\begin{array}{ccc} \mathsf{CMS} & \xrightarrow{H'} & \mathsf{CMS} \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathsf{Set} & \xrightarrow[H]{} & \mathsf{Set} \end{array}$$

for $U : \mathsf{CMS} \longrightarrow \mathsf{Set}$ the forgetful functor. Then H is iteratable and $U \cdot \mathcal{T}(H') = \mathcal{T}(H) \cdot U$. In fact, this follows from the results of [ARe] since U preserves limits. Any polynomial functor H on Set has a contracting lifting to CMS. For $HX = X^n$, define $H'(X, d) = (X, \frac{1}{2} d_{\max})$ (where $d_{\max}$ is the maximum metric) which is a contracting functor with $\varepsilon = \frac{1}{2}$. And coproducts of $\frac{1}{2}$-contracting liftings are $\frac{1}{2}$-contracting liftings of coproducts. The final coalgebra $\mathcal{T}(H')X$ is the Σ-tree algebra $T_\Sigma X$ equipped with a suitable complete metric. We will provide details of this metric later, see Remark 7.15.

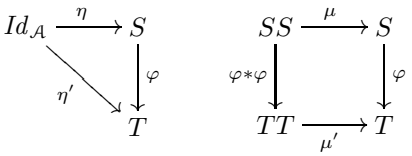
2.2. **Monads.** A *monad* on a category $\mathcal{A}$ is a triple (T, μ, η) consisting of a functor $T : \mathcal{A} \longrightarrow \mathcal{A}$, and natural transformations $\mu : TT \longrightarrow T$, and $\eta : Id \longrightarrow T$, satisfying the *unit laws* $\mu \cdot T\eta = \mu \cdot \eta T = id$, and the

associative law $\mu \cdot T\mu = \mu \cdot \mu T$:

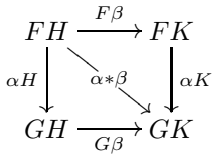


For a functor $H : \mathcal{A} \longrightarrow \mathcal{A}$ and a natural transformation $\alpha : F \longrightarrow G$ we use the usual notations $H\alpha$ and αH to denote the natural transformations with components $H(\alpha_X)$ and α_{HX}, respectively. Also it is customary to write just T for the monad in lieu of the triple, and we will follow this convention.

Let (S, η, μ) and (T, η', μ') be monads on the same category $\mathcal{A}$. A *morphism of monads* φ from S to T is a natural transformation $\varphi : S \longrightarrow T$ such that $\varphi \cdot \eta = \eta'$, and $\varphi \cdot \mu = \mu' \cdot (\varphi * \varphi)$:



This operation $*$ here is the *parallel composition* of natural transformations. In general, if $\alpha : F \longrightarrow G$ and $\beta : H \longrightarrow K$ are natural transformations, $\alpha * \beta : FH \longrightarrow GK$ is $\alpha K \cdot F\beta = G\beta \cdot \alpha H$.



We will denote by

$$\mathbf{Mon}(\mathcal{A})$$

the category of monads on $\mathcal{A}$ and their morphisms.

Example 2.6. Let H_Σ be a signature functor on Set. We already know how to define an object assignment T_Σ. In fact, T_Σ is a functor. We also have a natural transformation $\eta : Id \longrightarrow T_\Sigma$ which for each set X regards the elements of X as elements of $T_\Sigma X$. We additionally define a natural transformation $\mu_X : T_\Sigma(T_\Sigma X) \longrightarrow T_\Sigma X$, the operation which takes trees over trees over X into trees over X in the obvious way. In this way, we have a monad (T_Σ, μ, η) on Set.

Example 2.7. Let H be iteratable on a category $\mathcal{A}$. It has been shown in the previous work [AAMV, Mi$_2$] that the object assignment T (assigning to each object X the final coalgebra for $H(\_) + X$) gives rise to a monad on $\mathcal{A}$. This monad is characterized by a universal property—it is the free completely iterative monad on H. We will discuss this in detail later in Sections 3.4 and 4 below.

2.3. **Recursive Program Schemes.** We shall now explain how to capture recursive program schemes in a category-theoretic way. In order to do this we use a well-known adjunction between the category of signatures and the category of endofunctors of Set. For two categories $\mathcal{A}$ and $\mathcal{B}$ we denote by

$$[\mathcal{A}, \mathcal{B}]$$

the category of functors from $\mathcal{A}$ to $\mathcal{B}$.

Let Σ be a signature, i. e., $\Sigma : \mathbb{N} \longrightarrow \mathsf{Set}$ is a functor where $\mathbb{N}$ is regarded as a discrete category. Let $J : \mathbb{N} \longrightarrow \mathsf{Set}$ be the functor which maps a natural number n to the set $\{0, \ldots, n-1\}$. Recall that the functor $(\_) \cdot J : [\mathsf{Set}, \mathsf{Set}] \longrightarrow [\mathbb{N}, \mathsf{Set}]$ of restriction to $\mathbb{N}$ has a left-adjoint $\mathrm{Lan}_J(\_)$, i. e. the functor assigning to a signature its left Kan extension along J. Since $\mathbb{N}$ is a discrete category, the usual coend formula for computing left Kan extensions, see e. g. [ML], Theorem X.4.1, specializes to the coproduct in (2.1) above. That is, $\mathrm{Lan}_J(\Sigma)$ is the signature functor H_Σ. By virtue of the adjunction $\mathrm{Lan}_J(\_) \dashv (\_) \cdot J$ there is for any signature Σ and any endofunctor G of Set a bijection between natural transformations $\Sigma \longrightarrow G \cdot J$ and natural transformation $H_\Sigma \longrightarrow G$, and this bijection is natural in Σ and G. In fact, for any natural transformation $\alpha : \Sigma \longrightarrow G \cdot J$, i. e., a family of maps $\alpha_n : \Sigma_n \longrightarrow G\{0, \ldots, n-1\}$, we obtain a natural transformation $\beta : H_\Sigma \longrightarrow G$ as follows. The component β_X maps an element $(f, \vec{x})$ of $H_\Sigma X$ to $Gs(\alpha_n(f))$, where we consider the n-tuple $\vec{x}$ as a function $s : Jn \longrightarrow X$. Conversely, given $\beta : H_\Sigma \longrightarrow G$ define α by $\alpha_n(f) = \beta_{Jn}(f, i_n)$, where i_n is the n-tuple $(0, \ldots, n-1)$. It is easy to see that the two constructions yield natural transformations, are mutually inverse and natural in Σ and G.

We shall now use the above bijective correspondence to express recursive program schemes as natural transformations. Suppose we have a signature Σ of given operation symbols. Let Φ be a signature of new operation symbols. Classically a *recursive program scheme* (or shortly, *RPS*) gives for each operation symbol $f \in \Phi_n$ a term t^f over $\Sigma + \Phi$ in n variables. We thus have a system of formal equations

$$f(x_1, \ldots, x_n) \approx t^f(x_1, \ldots, x_n), \qquad f \in \Phi_n, \qquad n \in \mathbb{N}\,. \tag{2.3}$$

Now observe that the names of the variables in (2.3) do not matter. More precisely, regarding Φ an a functor from $\mathbb{N}$ to Set, any RPS as in (2.3) gives rise to a natural transformation

$$\Phi \longrightarrow T_{\Sigma+\Phi} \cdot J\,. \tag{2.4}$$

The formulation in (2.4) insures that in each equation of an RPS such as 2.3, if the symbol on the left side is n-ary, then the variables that can appear on the right are the n elements of $\{\, 0, \ldots, n-1 \,\}$. Notice as well that our formulation extends the classical notion of RPS in the sense that by taking $T_{\Sigma+\Phi}$ we allow infinite trees on the right-hand sides. Furthermore, we will generalize this notion of RPS.

The natural transformation in (2.4) corresponds to a unique natural transformation

$$H_\Phi \longrightarrow T_{\Sigma+\Phi}\,. \tag{2.5}$$

as explained above. The point is that the formulation in (2.5) is more useful to us than the one in (2.4) because (2.5) involves a natural transformations between endofunctors on one and the same category.

Now notice that $T_{\Sigma+\Phi} = \mathcal{T}(H_\Sigma + H_\Phi)$, where H_Σ and H_Φ denote the signature functors. With this in mind, we can rewrite (2.5), and we see that recursive program schemes correspond to natural transformations of the following form:

$$H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)\,.$$

This explains the work we have done so far.

To summarize: we abstract away from signatures and sets and study the uninterpreted and the interpreted semantics of recursive program schemes considered as natural transformations of the form

$$V \longrightarrow \mathcal{T}(H + V),$$

where H, V and $H + V$ are iteratable endofunctors on the category $\mathcal{A}$. Now to say what a solution of such a recursive program scheme is we first need to have a notion of (generalized) second-order substitution, see [C] for the classical notion. It turns out that the universal property of the free completely iterative monads $\mathcal{T}(H)$ readily yields this desired generalization. And this is the reason we are interested in monads in this paper.

Example 2.8. Let Σ contain a unary operation symbol G and a binary one F. The signature Φ of recursively defined operations contains two unary symbols φ and ψ. Consider the recursive program scheme (1.1). The signature functor expressing the givens is $H_\Sigma = X + (X \times X)$ and the recursively defined operations Φ are expressed by $H_\Phi X = X + X$. Thus, the scheme (1.1) gives a natural transformation $H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$. Similarly, the RPS (1.2) defining the factorial function with the signature Σ of givens and the signature Φ containing only the unary operation symbol f gives rise to a natural transformation $H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$, where $H_\Sigma X = 1 + X + X \times X$ and $H_\Phi X = X$.

2.4. **Eilenberg-Moore Algebras.** Recall that if (F, G, η, ϵ) is an adjunction, we get the associated monad on the domain of F by taking $T = GF$, $\mu = G\epsilon F$, and η from the adjunction. We also need a converse of this result. Given a monad T on $\mathcal{A}$, the *Eilenberg-Moore* category $\mathcal{A}^T$ of T has as objects the (monadic) T*-algebras*: these are morphisms $\alpha : TA \longrightarrow A$ such that the diagrams below commute:

$$\begin{array}{ccc} A & \xrightarrow{\eta_A} & TA \\ & \searrow & \downarrow{\scriptstyle \alpha} \\ & & A \end{array} \qquad \text{and} \qquad \begin{array}{ccc} TTA & \xrightarrow{\mu_A} & TA \\ {\scriptstyle T\alpha}\downarrow & & \downarrow{\scriptstyle \alpha} \\ TA & \xrightarrow[\alpha]{} & A\,. \end{array}$$

A morphism from $\alpha : TA \longrightarrow A$ to $\beta : TB \longrightarrow B$ is a morphism $h : A \longrightarrow B$ in $\mathcal{A}$ such that the square

$$\begin{array}{ccc} TA & \xrightarrow{\alpha} & A \\ {\scriptstyle Th}\downarrow & & \downarrow{\scriptstyle h} \\ TB & \xrightarrow[\beta]{} & B \end{array}$$

commutes. We usually write T-algebras using the notation of pairs, as in (A, α).

The relation between this construction and monads is that for any monad T, there is an adjunction $(F^T, U^T, \eta, \epsilon)$ from $\mathcal{A}$ to $\mathcal{A}^T$ to which T is associated. Here F^T is the functor taking A to the free T-algebra $\mu_A : TTA \longrightarrow TA$; $U^T : \mathcal{A}^T \longrightarrow \mathcal{A}$ is the forgetful functor taking the T-algebra $\alpha : TA \longrightarrow A$ to its carrier A; and in the same notation, $\epsilon_{(A,\alpha)}$ is α itself, taken to be a morphism of T-algebras, see [ML], Section VI.2.

3. Completely Iterative Algebras and Complete Elgot algebras

For interpreted solutions of recursive program schemes we need a suitable notion of algebras which can serve as interpretation of the givens. By a "suitable algebra" we mean, of course, one in which recursive program schemes have a *solution*. For example, for the recursive program scheme in (1.1) we are interested in those Σ-algebras, Σ a signature with a binary symbol F and a unary one G, in which we can obtain two new operations φ_A, ψ_A on A so that the formal equations of (1.1) become valid identities in A. In the classical theory one works with continuous algebras; i. e., algebras carried by a cpo such that all operations are continuous maps. Alternatively, one can work with algebras carried by a complete metric space such that all operations are contracting maps. In both of these approaches one imposes extra structure on the algebra in a way that makes it possible to obtain the semantics of a recursive definition as a join (or limit, respectively) of finite approximations.

When analyzing the two above types of algebras it turns out that they share a crucial feature that allows for the solution of recursive program schemes: these algebras induce an *evaluation* of all Σ-trees. More precisely, we consider a Σ-algebra A with a canonical map $T_\Sigma A \longrightarrow A$ providing for each Σ-tree over A its evaluation in A. It seems to us that in order to be able to obtain solutions of recursive program schemes in a Σ-algebra the minimal requirement is the existence of such an evaluation map turning A into an Eilenberg-Moore algebra of the monad T_Σ, see Example 2.6. More generally, we work here with *complete Elgot algebras* for an iteratable endofunctor H, which turn out to be precisely the Eilenberg-Moore algebras for the monad $\mathcal{T}(H)$, see [AMV$_3$]. An important subclass of all complete Elgot algebras are *completely iterative algebras* [Mi$_2$]. One of our main results (Theorem 7.3(ii)) states that recursive program schemes have unique solutions in completely iterative algebras.

Let us begin by explaining the notion of completely iterative algebra with an example. Let Σ be a signature, and let Y be any set. We think of Y as a set of parameters. The Σ-algebra $T_\Sigma Y$ of all (finite and infinite) Σ-trees over Y allows for the unique solution of *flat systems* of equations, i. e., systems of formal equations

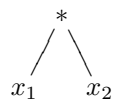
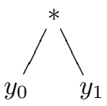
$$x_i \approx t_i \qquad i \in I, \tag{3.1}$$

with (recursion) variables from a set $X = \{ x_i \mid i \in I \}$, where either $t_i \in T_\Sigma Y$ is a Σ-tree with no recursion variables or else $t_i = \sigma(x_1, \ldots, x_n)$, $\sigma \in \Sigma_n$, $x_1, \ldots, x_n \in X$, is a flat Σ-tree.

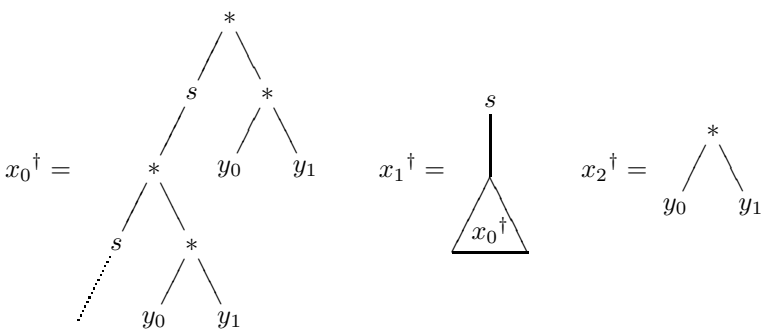
We have already begun in this paper to use the standard practice of using $\approx$ in a system to denote formal equations (recursive specifications of functions or other objects). We use $=$ to denote actual identity (see just below for an example). Flat systems have a unique *solution*: there exists a unique tuple ${x_i}^\dagger$, $i \in I$, of trees in $T_\Sigma Y$ such that the identities

$${x_i}^\dagger = t_i[\, x_i := {x_i}^\dagger \,]_{i \in I}$$

hold. For example, let Σ consist of a binary operation symbol $*$ and a unary one s. The following flat system of equations

$x_0 \approx$  $x_1 \approx$  $x_2 \approx$ 

with variables $X = \{ x_1, x_2, x_3 \}$ and parameters $Y = \{ y_0, y_1 \}$ has as its unique solution the following trees in $T_\Sigma Y$:



Observe that to give a flat system of equations is the same as to give a mapping $e : X \longrightarrow H_\Sigma X + T_\Sigma Y$ and a solution is the same as a mapping $e^\dagger : X \longrightarrow T_\Sigma Y$ such that the following square

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger \quad} & T_\Sigma Y \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\tau, T_\Sigma Y]} \\ H_\Sigma X + T_\Sigma Y & \xrightarrow[H_\Sigma e^\dagger + T_\Sigma Y]{} & H_\Sigma T_\Sigma Y + T_\Sigma Y \end{array}$$

commutes, where τ denotes the tree-tupling map. (We adopt the convention of denoting in a commutative diagram the identity on an object by that object itself.)

3.1. **Completely Iterative Algebras.** The example above suggests the following definition, originally studied in [Mi$_2$].

Definition 3.1. Let $H : \mathcal{A} \longrightarrow \mathcal{A}$ be an endofunctor. By a *flat equation morphism* in an object A (*of parameters*) we mean a morphism

$$e : X \longrightarrow HX + A\,.$$

Let $a : HA \longrightarrow A$ be an H-algebra. We say that $s : X \longrightarrow A$ is a *solution* of e in A if the square below commutes:

$$\begin{array}{ccc} X & \xrightarrow{\quad s \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha, A]} \\ HX + A & \xrightarrow[Hs + A]{} & HA + A \end{array} \tag{3.2}$$

We call A a *completely iterative algebra* (or *cia*, for short) if every flat equation morphism in A has a unique solution.

Observe that we have no restriction on our objects of variables. (That is, in the case of Set, we do not require that the set of variables be finite.) Imposing this restriction weakens the notion to what [AMV$_1$] calls an *iterative algebra*. It will be essential in this paper to consider equation morphisms whose domain is not finite.

Examples 3.2.

(i) Let $\mathcal{P}_\mathrm{f}$ be the finite power set functor on Set, and assume the Anti-Foundation Axiom. Let HF_1 be the set of hereditarily finite sets. Let τ be the inclusion of $\mathcal{P}_\mathrm{f}(HF_1)$ into HF_1. This map τ turns HF_1 into a cia with respect to $\mathcal{P}_\mathrm{f}$.

(ii) Consider the subalgebra $HF_{1/2}$ of sets whose transitive closure is finite, then the *complete* iterativity is lost. Only finite systems can be solved in this setting. For more on this example and the last, see Section 18.1 of [BM].

(iii) Final coalgebras. In [Mi$_2$] it is proved that for a final H-coalgebra $\alpha : T \longrightarrow HT$ the inverse $\tau : HT \longrightarrow T$ of the structure map yields a cia. Analogously, for every object Y of $\mathcal{A}$ a final coalgebra TY of $H(\_) + Y$ yields a cia, see Theorem 3.12 below. This generalizes the first example and the examples $T_\Sigma Y$ of all Σ-trees over a set Y.

(iv) Let H be a contracting endofunctor on the category CMS of complete metric spaces, see Example 2.2(iii). Then any non-empty H-algebra (A, α) is completely iterative. In fact, given any flat equation morphism $e : X \longrightarrow HX + A$ in CMS, it is not difficult to prove that the assignment $s \longmapsto \alpha \cdot (Hs + A) \cdot e$ is a contracting function of $\mathsf{CMS}(X, A)$, see [AMV$_3$]. Then, by Banach's Fixed Point Theorem, there exists a unique fixed point of that contracting function, viz. a unique solution $e^\dagger$ of e. Notice that $e^\dagger$ is obtained as a limit of a Cauchy sequence. In fact, choose some element $a \in A$ and define the Cauchy sequence $(e^\dagger_n)_{n \in \mathbb{N}}$ in $\mathsf{CMS}(X, A)$ by recursion as follows: let $e^\dagger_0 = \mathsf{const}_a$, and given $e^\dagger_n$ define $e^\dagger_{n+1}$ by the commutativity of the square

$$\begin{array}{ccc} X & \xrightarrow{\quad e^\dagger_{n+1} \quad} & A \\ {\scriptstyle e}\downarrow & & \uparrow{\scriptstyle [\alpha, A]} \\ HX + A & \xrightarrow[He^\dagger_n + A]{} & HA + A \end{array} \tag{3.3}$$

(v) Non-empty compact subsets form cias. Let (X,d) be a complete metric space. Consider the set $C(X)$ of all non-empty compact subspaces of X together with the so-called Hausdorff metric h; for two compact subsets A and B of X the distance $h(A,B)$ is the smallest number r such that B can be covered by open balls of radius r around each point of A, and vice versa, A can be covered by such open balls around each point of B. In symbols, $h(A,B) = \max\{\, d(A \to B), d(B \to A)\,\}$, where $d(A \to B) = \max_{a\in A} \min_{b\in B} d(a,b)$. It is well-known that $(C(X),h)$ forms a complete metric space; see, e.g. [B]. Furthermore, if $f_i : X \longrightarrow X$, $i = 1,\ldots,n$, are contractions of the space X with contraction factors c_i, $i = 1,\ldots,n$, then it is easy to show that the map

$$\alpha_X : C(X)^n \longrightarrow C(X) \qquad (A_i)_{i=1,\ldots,n} \longmapsto \bigcup_{i=1}^{n} f_i[A_i]$$

is a contraction with contraction factor $c = \max_i c_i$ (the product $C(X)^n$ is, of course, equipped with the maximum metric). In other words, given the f_i, we obtain on $C(X)$ the structure α_X of an H-algebra of the contracting endofunctor $H(X,d) = (X^n, c\cdot d_{\max})$. Thus, if there exists a non-empty compact subset of X, then $(C(X),\alpha_X)$ is a cia.

As an illustration we show that the Cantor "middle third" set s may be obtained via the cia structure on a certain space. Recall that s is the unique non-empty closed subset of the interval $I = [0,1]$ which satisfies $s = \frac{1}{3}s \cup (\frac{2}{3} + \frac{1}{3}s)$. (We write $\frac{1}{3}s$ to mean $\{\, \frac{x}{3} \mid x \in s\,\}$, of course.) So let (X,d) be the Euclidean interval $I = [0,1]$ and consider the $\frac{1}{3}$-contracting functions $f(x) = \frac{1}{3}x$ and $g(x) = \frac{1}{3}x + \frac{2}{3}$ on I. Then $\alpha_I : C(I)\times C(I) \longrightarrow C(I)$ with $\alpha_I(A,B) = f[A]\cup g[B]$ gives the structure of a cia on $C(I)$ of the functor $H(X,d) = (X\times X, \frac{1}{3}\cdot d_{\max})$, which is a lifting of the signature functor $H_\Sigma X = X\times X$ of Set expressing one binary operation symbol α. Now consider the formal equation

$$x \approx \alpha(x,x)\,.$$

It gives rise to a flat equation morphism $e : 1 \longrightarrow H1 + C(I)$ which maps the element of the trivial one point space 1 to the element of $H1 = 1$. The unique solution $e^\dagger : 1 \longrightarrow C(I)$ picks a non-empty closed set s satisfying $s = \alpha(s,s) = f[s]\cup g[s]$, i. e., $e^\dagger$ picks the Cantor set.

(vi) Continuing with our last point, for each non-empty closed $t \in C(I)$, there is a unique $s = s(t)$ with $s = \alpha(s,t)$. The argument is just as above. But the work we have done does *not* show that the map $t \longmapsto s(t)$ is continuous. For this, we would have to study a recursive program scheme $\varphi(x) \approx \alpha(\varphi(x),x)$ and solve this in $(C(I),\alpha_I)$ in the appropriate sense. Our work later in the paper does exactly this, and it follows that the solution to $\varphi(x) \approx \alpha(\varphi(x),x)$ in the given algebra is the *continuous* function $t \longmapsto s(t)$.

(vii) Suppose that $H : \mathsf{Set} \longrightarrow \mathsf{Set}$ has a lifting to a contracting endofunctor H' on CMS, see Example 2.5(vi). Let $\alpha : HA \longrightarrow A$ be an H-algebra. If there exists a complete metric, say d, on A such that α is a nonexpanding map $H'(A,d) \longrightarrow (A,d)$, then A is a completely iterative algebra: to every equation morphism $e : X \longrightarrow HX+A$ assign the unique solution of $e : (X,d_0) \longrightarrow H'(X,d_0)+(A,d)$, where d_0 is the discrete metric on X; i. e., $d(x,x') = 1$ iff $x \neq x'$.

3.2. **Complete Elgot Algebras.** In many settings, one studies a fixed point operation on a structure like a complete partial order. And in such settings, one typically does not have *unique* fixed points. So completely iterative algebras are *not* the unifying concept capturing precisely what is needed to solve recursive program schemes. Instead, we shall need a weakening of the notion of a cia.

Remark 3.3. We will need two operations in the statement of Definition 3.4 below. The first operation formalizes the renaming of parameters in a flat equation morphism. More precisely, for a flat equation morphism $e : X \longrightarrow HX + A$ and a morphism $h : A \longrightarrow B$ we define

$$h \bullet e \equiv X \xrightarrow{\ e\ } HX + A \xrightarrow{\ HX+h\ } HX + B\,.$$

The second operation allows us to combine two flat equation morphisms where the parameters of the first are the variables of the second into one "simultaneous" flat equation morphism. More precisely, given two flat equation morphisms $e : X \longrightarrow HX + Y$ and $f : Y \longrightarrow HY + A$ we define

$$f \blacksquare e \equiv X + Y \xrightarrow{\ [e,\mathsf{inr}]\ } HX + Y \xrightarrow{\ HX+f\ } HX + HY + A \xrightarrow{\ \mathsf{can}+A\ } H(X+Y) + A\,.$$

Definition 3.4. A complete *Elgot algebra* is a triple $(A,a,(\_)^\dagger)$, where (A,a) is an H-algebra, and $(\_)^\dagger$ assigns to every flat equation morphism e with parameters in A a solution $e^\dagger : X \longrightarrow A$ such that the following two laws hold:

Functoriality. Solutions respect renaming of variables. Given two flat equation morphisms e and f with parameters in A and a morphism h of equations between them; i. e., the square

$$\begin{array}{ccc} X & \xrightarrow{e} & HX + A \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh+A} \\ Y & \xrightarrow[f]{} & HY + A \end{array}$$

commutes. We then have

$$e^\dagger = f^\dagger \cdot h\,.$$

Compositionality. Simultaneous recursion may be performed sequentially. For all flat equation morphisms $e : X \longrightarrow HX + Y$ and $f : Y \longrightarrow HY + A$, the solution of the combined equation morphism $f \blacksquare e$ is obtained by first solving f and then solving e "plugging in" $f^\dagger$ for the parameters:

$$(f^\dagger \bullet e)^\dagger = (f \blacksquare e)^\dagger \cdot \mathsf{inl}\,.$$

Definition 3.5. A *homomorphism* h from an Elgot algebra $(A, a, (\_)^\dagger)$ to an Elgot algebra $(B, b, (\_)^\ddagger)$ (for the same functor H) is a morphism $h : A \longrightarrow B$ that preserves solutions, i. e., for every flat equation morphism $e : X \longrightarrow HX + A$ we have a commutative triangle

$$\begin{array}{ccc} & X & \\ {\scriptstyle e^\dagger}\swarrow & & \searrow{\scriptstyle (h\bullet e)^\ddagger} \\ A & \xrightarrow[h]{} & B\,. \end{array}$$

Proposition 3.6. [AMV$_3$] *Every homomorphism $h : (A, a, (\_)^\dagger) \longrightarrow (B, b, (\_)^\ddagger)$ of Elgot algebras is a homomorphism of H-algebras; i. e., the square*

$$\begin{array}{ccc} HA & \xrightarrow{a} & A \\ {\scriptstyle Hh}\downarrow & & \downarrow{\scriptstyle h} \\ HB & \xrightarrow[b]{} & B \end{array}$$

commutes. The converse is false in general. If, however, A and B are cias, then any H-algebra morphism is a homomorphism of Elgot algebras.

Proof. We sketch the argument of the first statement, omitting some of the details. First, consider the flat equation morphism

$$e_A \equiv HA + A \xrightarrow{H\mathsf{inr}+A} H(HA + A) + A\,.$$

Its solution is $e_A^\dagger = [a, A] : HA + A \longrightarrow A$ as one easily establishes using the commutativity of Diagram 3.2 for $e_A^\dagger$. Similarly, we have $e_B : HB + B \longrightarrow H(HB + B) + B$ with $e_B^\ddagger = [b, B]$. Now consider h as in our proposition. Then the equation $(h \bullet e_A)^\ddagger = h \cdot e_A^\dagger$ holds. Furthermore, $Hh + h : HA + A \longrightarrow HB + B$ is a morphism of equations from $h \bullet e_A$ to e_B, thus Functoriality yields $(h \bullet e_A)^\ddagger = e_B^\ddagger \cdot (Hh + h)$. Now one readily performs the computation

$$[h \cdot a, h] = h \cdot e_A^\dagger = (h \bullet e_A)^\ddagger = e_B^\ddagger \cdot (Hh + h) = [b \cdot Hh, h]\,.$$

The desired equation $h \cdot a = b \cdot Hh$ follows from the left-hand component. □

Remark 3.7.

(i) In [AMV$_3$] there is also a notion of a non-complete Elgot algebra. Since we will only be interested in using complete Elgot algebras we will henceforth understand by an Elgot algebra a complete one.

(ii) The axioms of Elgot algebras are inspired by the axioms of iteration theories of Bloom and Ésik [BE]. In fact, the two laws above are essentially "flat" versions of the commutative identities and the left pairing identity (also known as Bekić-Scott law) from [BE].

One justification for the above axioms is that Elgot algebras turn out to be the Eilenberg-Moore category of the monad T, see Subsection 3.5. We shall mention this result at the end of this section. Applied to a signature functor H_Σ on Set, that means a Σ-algebra A is an Elgot algebra precisely if there exists a canonical map $T_\Sigma A \longrightarrow A$ evaluating all Σ-trees in A.

(iii) Notice, that flat equation morphisms are precisely the coalgebras of the functor $H(\_) + A$. Thus, the Functoriality above states that $(\_)^\dagger$ is a functor from the category of flat equation morphisms to the comma category $\mathcal{A}/A$.

Examples 3.8. We present some examples of Elgot algebras. None but the first are in general cias.

(i) Completely iterative algebras are Elgot algebras. It is proved in [AMV$_3$] that for a cia the assignment of the unique solution to any flat equation morphism satisfies the Functoriality and the Compositionality.

(ii) Continuous algebras. Let H be a locally continuous endofunctor on CPO, see Example 2.2(ii). It is shown in [AMV$_3$] that any H-algebra (A, a) with a least element $\bot$ is an Elgot algebra when to a flat equation morphism $e : X \longrightarrow HX + A$ the least solution $e^\dagger$ is assigned. More precisely, define $e^\dagger$ as the join of the following increasing ω-chain in $\mathsf{CPO}(X, A)$: $e^\dagger_0$ is the constant function $\bot$; and given $e^\dagger_n$ let $e^\dagger_{n+1} = [a, A] \cdot (He^\dagger_n + A) \cdot e$, so that Diagram (3.3) commutes.

(iii) Suppose that $H : \mathsf{Set} \longrightarrow \mathsf{Set}$ is a functor with a locally continuous lifting $H' : \mathsf{CPO} \longrightarrow \mathsf{CPO}$. An H-algebra $\alpha : HA \longrightarrow A$ is called CPO-*enrichable* if there exists a complete partial order $\sqsubseteq$ on A such that A becomes a continuous algebra $\alpha : H'(A, \sqsubseteq) \longrightarrow (A, \sqsubseteq)$ with a least element. Any CPO-enrichable algebra A is an Elgot algebra: to every equation morphism $e : X \longrightarrow HX + A$, let $\leq$ be the discrete order on X, consider $\widehat{e} : (X, \leq) \longrightarrow H'(X, \leq) + (A, \sqsubseteq)$ defined in the obvious way, and assign $U\widehat{e}^\dagger : X \longrightarrow A$, where $\widehat{e}^\dagger$ is from part (ii), and $U : \mathsf{CPO} \longrightarrow \mathsf{Set}$ is the forgetful functor.

(iv) Every complete lattice A is an Elgot algebra of the endofunctor $HX = X \times X$ of Set. In fact, taking binary joins yields an H-algebra structure $\vee : A \times A \longrightarrow A$. Furthermore, observe that the algebra TA of all binary trees over A has an evaluation $\alpha : TA \longrightarrow A$ mapping every binary tree in TA to the join of its leaf labels. For any flat equation morphism $e : X \longrightarrow X \times X + A$ form the flat equation morphism $\eta_A \bullet e : X \longrightarrow X \times X + TA$ take its unique solution $(\eta_A \bullet e)^\dagger : X \longrightarrow TA$ and let $e^* = \alpha \cdot (\eta_A \bullet e)^\dagger$. Then $(A, a, (\_)^*)$ is an Elgot algebra for H, see [AMV$_3$], Example 3.9. Notice that this is usually not a cia since the formal equation $x \approx x \vee x$ has in general many different solutions in a complete lattice.

3.3. **Computation Tree Elgot Algebras.** In this section we present Elgot algebras for a signature that uses undefined elements and also conditionals. Let Σ be a signature, and let $H = H_\Sigma$ be the associated endofunctor on Set. Let (A, a) be any H_Σ-algebra, and let $\uparrow$ be any element of A. We shall define an Elgot algebra structure on A related to the natural computation tree semantics of recursive definitions, where solutions are obtained by rewriting. The idea is that $\uparrow$ is our "scapegoat" for ungrounded definitions.

We shall assume that the algebra A interprets the function symbols in Σ in a strict fashion: if any argument a_i is $\uparrow$, then the overall value $g_A(a_1, \ldots, a_n)$ is $\uparrow$ as well. Conversely, if $g_A(a_1, \ldots, a_n) = \uparrow$, we require that ($n \geq 1$ and) some a_i is $\uparrow$. We make this assumption for all function symbols g *except for* the *conditional symbol* $\mathsf{ifzero} \in \Sigma_3$. We make a different assumption on ifzero. For this, fix an element $0 \in A$ other than $\uparrow$. We want

$$\mathsf{ifzero}_A(x, y, z) = \begin{cases} y & \text{if } x = 0 \\ z & \text{if } x \neq 0 \text{ and } x \neq \uparrow \\ \uparrow & \text{otherwise} \end{cases} \tag{3.4}$$

To summarize, in this section we work with algebras of signature functors on Set which come with designated objects $\uparrow$ and 0 satisfying the assumptions above.

We shall work with partial functions and we use some notation which is standard. For partial functions $p, q : X \rightharpoonup A$, $p(x)\uparrow$ means that p is not defined on x, and we write $p(x)\downarrow$ if p is defined on x. Finally, $p(x) \simeq p(y)$ means that if either $p(x)$ or $q(y)$ is defined, then so is the other; and in this case, the values are the same.

Definition 3.9. Let $e : X \longrightarrow HX + A$ be a flat equation morphism in A. We define a partial function $\hat{e} : X \rightharpoonup A$ as follows:

(i) If $e(x) = a$ and $a \neq \uparrow$, then $\hat{e}(x) \simeq a$.
(ii) If $e(x) = g(x_1, \ldots, x_k)$, $g \neq \mathsf{ifzero}$, and for each i, $\hat{e}(x) \simeq a_i$, then $\hat{e}(x) \simeq g_A(a_1, \ldots, a_k)$.
(iii) If $e(x) = \mathsf{ifzero}(y, z, w)$ and $\hat{e}(y) \simeq 0$, then $\hat{e}(x) \simeq \hat{e}(z)$.
(iv) If $e(x) = \mathsf{ifzero}(y, z, w)$ and $\hat{e}(y)\downarrow$ but $\hat{e}(y) \not\simeq 0$, then $\hat{e}(x) \simeq \hat{e}(w)$.

We call $\hat{e}$ the *computation function corresponding to* e.

We intend this to be a definition of a partial function by recursion, so that we may prove facts about $\hat{e}$ by induction. Here is a first example, one which will be important in Proposition 3.10 below: if $\hat{e}(x)\downarrow$, then $\hat{e}(x) \neq \uparrow$.

Now that we have $\hat{e}$, we define $e^\dagger(x)$ to be $\hat{e}(x)$ if $\hat{e}$ is defined; if it is not, we set $e^\dagger(x) = \uparrow$. (Note that $e^\dagger(x) = \uparrow$ iff $\hat{e}(x)\uparrow$.)

In the statement of the result below, we also mention the main way that one obtains structures which satisfy the standing hypotheses of this section.

Proposition 3.10. *Let $A_0 = (A_0, a_0)$ be any H_Σ-algebra, let $0 \in A_0$, let $\uparrow \notin A_0$, and let $A = A_0 \cup \{\uparrow\}$. Let $A = (A, a)$ be defined in terms of this data by extending a_0 to the function $a : H_\Sigma A \longrightarrow A$ strictly on all function symbols except* ifzero*, and with* ifzero_A *given by (3.4). Let $(\_)^\dagger$ be as above. Then $(A, a, (\_)^\dagger)$ is an Elgot algebra.*

Proof. Clearly the assumption of this section hold for the algebra A. These assumptions ensure that A is CPO-enrichable. In fact, equip A with the flat cpo structure with the least element $\uparrow$. Then all operations on A are easily checked to be monotone, whence continuous; thus, $a : H_\Sigma A \longrightarrow A$ is a continuous algebra. By Example 3.8(iii), we obtain an Elgot algebra $(A, a, (\_)^*)$. We will prove that for any flat equation morphism $e : X \longrightarrow HX + A$ the least solution e^* agrees with the map $e^\dagger$ given by the computation function $\hat{e}$. To this end recall first that the set $\mathsf{Par}(X, A)$ of partial functions from X to A is a cpo with the order given by $f \sqsubseteq g$ if for all $x \in X$, $f(x)\downarrow$ implies $g(x)\downarrow$ and $f(x) = g(x)$. Now observe that the definition of $\hat{e}$ by recursion means that $\hat{e}$ is the join of an increasing chain in $\mathsf{Par}(X, A)$. In fact, let $\hat{e}_0$ be the everywhere undefined function and given $\hat{e}_n$ define $\hat{e}_{n+1}$ as follows: in the clauses (i)–(iv) in Definition 3.9 replace the term $\hat{e}(x)$ by $\hat{e}_{n+1}(x)$ and replace all occurrences of $\hat{e}$ on right-hand sides of defining equations by $\hat{e}_n$. Clearly, $(\hat{e}_n)_{n<\omega}$ is an increasing chain in $\mathsf{Par}(X, A)$ whose join is $\hat{e}$.

Now recall from Example 3.8(ii) that e^* is the join of the chain e^*_n in $\mathsf{CPO}(X, A)$, where X is discretely ordered. We shall show by induction that for every $x \in X$ the equation

$$e^*_n(x) = \begin{cases} \hat{e}_n(x) & \text{if } \hat{e}_n(x)\downarrow \\ \uparrow & \text{else} \end{cases} \tag{3.5}$$

holds. The base case is obvious. For the induction step we proceed by case analysis. If $e(x) = a$, then $e^*_{n+1}(x) = a$ and so (3.5) holds. The second case is $e(x) = g(x_1, \ldots, x_k)$, $g \neq \mathsf{ifzero}$. We have $e^*_{n+1}(x) = g_A(e^*_n(x_1), \ldots, e^*_n(x_k))$. By our assumptions, this equals $\uparrow$ precisely if at least one of the $e^*_n(x_j)$ is $\uparrow$, which in turn holds if and only if $\hat{e}_n(x_k)\uparrow$ for some j; and equivalently, $\hat{e}_{n+1}(x)\uparrow$. Otherwise all $\hat{e}_n(x_j)$ are defined and by induction hypothesis we get

$$e^*_{n+1}(x) = g_A(e^*_n(x_1), \ldots, e^*_n(x_k)) = g_A(\hat{e}_n(x_1), \ldots, \hat{e}_n(x_k)) = \hat{e}_{n+1}(x)\,.$$

Thirdly, assume that $e(x) = \mathsf{ifzero}(y, z, w)$. Then similarly as before we have

$$e^*_{n+1}(x) = \mathsf{ifzero}(e^*_n(y), e^*_n(z), e^*_n(w))\,.$$

We obtain $e^*_{n+1}(x) = \uparrow$ whenever $e^*_n(y) = \uparrow$. But this happens precisely if $\hat{e}_n(y)\uparrow$, which implies that $\hat{e}_{n+1}(x)\uparrow$. Now if $e^*_n(y) \neq \uparrow$, then we have equivalently that $\hat{e}_n(y)\downarrow$. We treat here only the case that $\hat{e}_n(y) = 0$; the remaining case is similar. In our present case it follows that $e^*_{n+1}(x) = e^*_n(z)$ and $\hat{e}_{n+1}(x) \simeq \hat{e}_n(z)$. Therefore, by the induction hypothesis, the desired equation (3.5) holds.

Finally, from (3.5) we conclude that for the least fixed points e^* and $\hat{e}$ we have

$$e^*(x) = \begin{cases} \hat{e}(x) & \text{if } \hat{e}(x)\downarrow \\ \uparrow & \text{else.} \end{cases}$$

Thus, we get $e^* = e^\dagger$ which completes the proof. □

Definition 3.11. Let $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$ be a signature functor, let $A_0 = (A_0, a_0)$ be any H_Σ-algebra as in the hypothesis of Proposition 3.10. We call the Elgot algebra $(A, a, (\_)^\dagger)$ the *computation tree Elgot algebra* induced by A_0.

We shall study the interpreted solutions of recursive program schemes in computation tree Elgot algebras in Section 7.1.

3.4. **Dramatis Personae.** As we mentioned already the classical theory of recursive program schemes rests on the fact that in any continuous algebra A all Σ-trees over A can be evaluated; i. e., there is a canonical map $T_\Sigma A \longrightarrow A$. In a suitable category of cpos the structures $T_\Sigma X$ play the rôle of *free algebras*. The freeness is used to define maps *out* of those algebras. In our setting, the Σ-trees are the final coalgebra. So in order to generalize the classical theory, we need a setting in which the final coalgebras TY are *free algebras*. The following result gives such a setting. It is fundamental for the rest of the paper and collects the results of Theorems 2.8 and 2.10 of [Mi_2] and Theorem 5.6 of [AMV_3]. We sketch a proof for the convenience of the reader.

Theorem 3.12. *Let H be any endofunctor of $\mathcal{A}$. The following are equivalent:*

(i) *TY is a final coalgebra of $H(\_) + Y$,*
(ii) *TY is a free completely iterative H-algebra on Y, and*
(iii) *TY is a free (complete) Elgot H-algebra on Y.*

In more detail: if (TY, α_Y) is a final coalgebra for $H(\_) + Y$, the inverse $[\tau_Y, \eta_Y] : HTY + Y \longrightarrow Y$ of α_Y gives a cia for H, which as an Elgot algebra is free on Y. Conversely, given a free Elgot H-algebra $(TY, \tau_Y, (\_)^\dagger)$ with a universal arrow $\eta_Y : Y \longrightarrow TY$, then this is a cia, whence a free cia on Y, and $[\tau_Y, \eta_Y]$ is an isomorphism whose inverse is the structure map of a final coalgebra for $H(\_) + Y$.

Sketch of Proof. Suppose first that (TY, α_Y) is a final coalgebra for $H(\_) + Y$. Let $[\tau_Y, \eta_Y]$ be the inverse of α_Y. Then $\tau_Y : HTY \longrightarrow TY$ is a completely iterative algebra for H, and therefore an Elgot algebra. In fact, for any flat equation morphism $e : X \longrightarrow HX + TY$, form the following coalgebra

$$c \equiv X + TY \xrightarrow{[e,\mathsf{inr}]} HX + TY \xrightarrow{HX+\alpha_Y} HX + HTY + Y \xrightarrow{\mathsf{can}+Y} H(X + TY) + Y$$

and define

$$e^\dagger \equiv X \xrightarrow{\mathsf{inl}} X + TY \xrightarrow{h} TY\,,$$

where h is the unique homomorphism from the coalgebra $(X + TY, c)$ to the final one. It is not difficult to prove that $e^\dagger$ is the unique solution of e.

Furthermore, $(TY, \tau_Y, (\_)^\dagger)$ is a free Elgot algebra on Y. For any Elgot algebra $(A, a, (\_)^\ddagger)$ and any morphism $m : Y \longrightarrow A$ form the equation morphism

$$m \bullet \alpha_Y \equiv TY \xrightarrow{\alpha_Y} HTY + Y \xrightarrow{HTY+m} HTY + A$$

It is shown in Theorem 5.8 of [AMV$_3$] that the solution $h = (m \bullet \alpha_Y)^\ddagger$ yields the unique homomorphism $h : TY \longrightarrow A$ of Elgot algebras such that $h \cdot \eta_Y = m$.

Now conversely, assume that $(TY, \tau_Y, (\_)^\dagger)$ is a free Elgot algebra on Y with a universal arrow $\eta_Y : Y \longrightarrow TY$. It can be shown that $[\tau_Y, \eta_Y]$ is an isomorphism, see Lemma 5.7 of [AMV$_3$]. Denote by α_Y its inverse. Then (TY, τ_Y) is a cia; i. e., for any flat equation morphism $e : X \longrightarrow HX + TY$ the solution $e^\dagger$ is unique. In fact, suppose that s is any solution of e. It follows that s is a morphism of equations from e to the flat equation morphism

$$f \equiv TY \xrightarrow{\alpha_Y} HTY + Y \xrightarrow{HTY+\eta_Y} HTY + TY\,.$$

Thus, $f^\dagger \cdot s = e^\dagger$ by Functoriality of $(\_)^\dagger$. Next one can show, using the Compositionality, that $f^\dagger : TY \longrightarrow TY$ is a homomorphism of Elgot algebras satisfying $f^\dagger \cdot \eta_Y = \eta_Y$. Thus, by the freeness of TY, $f^\dagger = id$. This proves that (TY, τ_Y) is a cia, which implies that it is the free one on Y. It is not difficult to show that this yields a final coalgebra of $H(\_) + Y$. In fact, for any coalgebra $c : C \longrightarrow HC + Y$ the unique solution of the flat equation morphism $\eta_Y \bullet c$ yields a unique homomorphism $(C, c) \longrightarrow (TY, \alpha_Y)$ of coalgebras. □

Theorem 3.12 has an important consequence for our work. Recall that we assume H is iteratable, so $H(\_) + Y$ *does* have a final coalgebra for all Y. The next result gives the *dramatis personae* for the rest of the paper.

Theorem 3.13. *There is a left adjoint to the forgetful functor from $\mathsf{Alg}^\dagger\, H$, the category of Elgot algebras and their homomorphisms, to the base category $\mathcal{A}$*

$$\mathsf{Alg}^\dagger\, H \underset{U}{\overset{L}{\underset{\longrightarrow}{\overset{\longleftarrow}{\bot}}}} \mathcal{A}\,.$$

The left-adjoint L assigns to each object Y of $\mathcal{A}$ a free Elgot algebra $(TY, \tau_Y, (\_)^\dagger)$ on Y (equivalently, (TY, α_Y) where $\alpha_Y = [\tau_Y, \eta_Y]^{-1}$ is a final coalgebra of $H(\_) + Y$). The unit of the adjunction is η whose components are given by the universal arrows $\eta_Y : Y \longrightarrow TY$ of the free Elgot algebras. The counit ε gives for each Elgot algebra $(A, a, (\_)^\ddagger)$ the unique homomorphism $\widetilde{a} : TA \longrightarrow A$ of Elgot algebras such that $\widetilde{a} \cdot \eta_A = id$. We have

$$\widetilde{a} = (\alpha_A)^\ddagger : TA \longrightarrow A\,, \tag{3.6}$$

where $\alpha_A : TA \longrightarrow HTA + A$ is considered as a flat equation morphism with parameters in A.

Moreover, we obtain additional structure:

(i) *A monad $(\mathcal{T}(H), \eta^H, \mu^H)$ on $\mathcal{A}$ such that for all objects Y of $\mathcal{A}$,*
  (a) *$\mathcal{T}(H)Y = TY$ is the carrier of a final coalgebra of $H(\_) + Y$;*

(b) μ^H_Y *is the (unique) solution of* α_{TY}*, considered as a flat equation morphism with parameters in* TY*.*

(ii) *A natural transformation* $\alpha^H : T \longrightarrow HT + Id$*.*

(iii) *A natural transformation* $\tau^H : HT \longrightarrow T$ *such that* $[\tau^H, \eta^H]$ *is the inverse of* α^H*.*

(iv) *A "canonical embedding"* κ^H *of* H *into* T*:*

$$\kappa^H \equiv H \xrightarrow{H\eta^H} HT \xrightarrow{\tau^H} T\,. \tag{3.7}$$

Proof. It is obvious that the assignment of Y to a free Elgot algebra on Y yields a left-adjoint to U. This adjunction gives rise to a monad $(\mathcal{T}(H), \eta^H, \mu^H)$ on $\mathcal{A}$ which assigns to every object of $\mathcal{A}$ the underlying object TY of a free Elgot algebra on Y, see Section 2.4. Thus item (a) follows from Theorem 3.12. The monad multiplication μ^H is given by $U\varepsilon L$, i. e., $\mu^H_Y : TTY \longrightarrow TY$ is the unique homomorphism of Elgot algebras such that $\mu^H_Y \cdot \eta^H_{TY} = id$. It follows from the proof of Theorem 3.12 that $\widetilde{a} = (m \bullet \alpha_A)^\ddagger$, where m is the identity on A. The special instance of this where A is the free Elgot algebra TY yields (b). The functoriality of T implies that the algebra structures τ_Y, and the coalgebra structures α_Y, Y in $\mathcal{A}$, form natural transformations. It follows from Theorem 3.12 that α^H and $[\tau^H, \eta^H]$ are mutually inverse. □

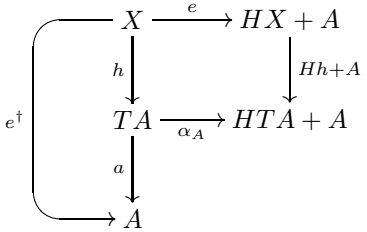
We call the monad $(\mathcal{T}(H), \eta^H, \mu^H)$ the *completely iterative monad* generated by H. (The name comes from an important property which we discuss in Section 4.) As always, we just write $\mathcal{T}(H)$, or even shorter T to denote this monad, and we shall frequently drop the superscripts when dealing with the structure coming from a single endofunctor H. (But as the reader will see later, we frequently do need to consider the structures coming from two endofunctors. This is particularly pertinent in our study since in recursive program schemes we usually have two signatures, hence two functors, see Section 2.3).

For any Elgot algebras $(A, a, (\_)^\dagger)$ for H we call the homomorphism $\widetilde{a} : TA \longrightarrow A$ in (3.6) above the *evaluation morphism* of that Elgot algebra. Theorem 3.14 below shows that Elgot algebras are equivalently presented by their evaluation morphisms.

3.5. **The Eilenberg-Moore Category of $\mathcal{T}(H)$.**

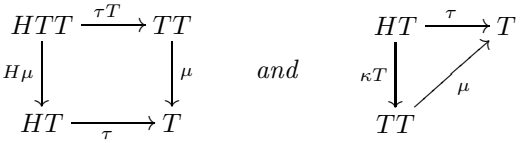
Theorem 3.14. [AMV$_3$] *The category* $\mathsf{Alg}^\dagger H$ *of Elgot algebras is isomorphic to the Eilenberg-Moore category* $\mathcal{A}^T$ *of monadic* T*-algebras. More precisely, for any Elgot algebra* $A = (A, a, (\_)^\dagger)$*, the evaluation morphism* $U\epsilon_A = \widetilde{a} : TA \longrightarrow A$ *is an Eilenberg-Moore algebra of* T*.*

Conversely, for any Eilenberg-Moore algebra $a : TA \longrightarrow A$ *we obtain an Elgot algebra by using as structure map the composite* $a \cdot \kappa_A : HA \longrightarrow A$*, and by defining for a flat equation morphism* $e : X \longrightarrow HX + A$ *the solution* $e^\dagger = a \cdot h$*, where* h *is the unique coalgebra homomorphism from* (X, e) *to* (TA, α_A)*:*

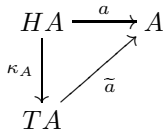


These two constructions extend to the level of morphisms, and they yield the desired isomorphism between the two categories $\mathsf{Alg}^\dagger H$ *and* $\mathcal{A}^T$*.*

Corollary 3.15. *The diagrams*



commute, and for every Elgot algebra $(A, a, (\_)^\dagger)$ *the triangle*



commutes.

Proof. The lower triangle commutes because the two constructions of Theorem 3.14 are mutually inverse. The special cases of this triangle for $A = TY$ and $a = \tau_Y$ yield the commutativity of the upper right-hand triangle since $\mu_Y = U\varepsilon_{TY} = \widetilde{\tau_Y}$. Finally, the upper left-hand square commutes since for each Y in $\mathcal{A}$, μ_Y is a homomorphism of Elgot algebras, whence an H-algebra homomorphism by Proposition 3.6. □

4. Completely Iterative Monads

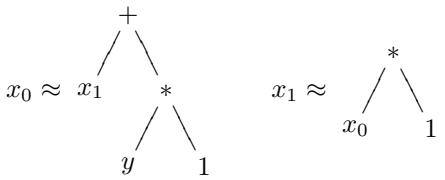
Before we can state a theorem providing solutions of (generalized) recursive program schemes, we need to explain what a solution is. In the classical setting one introduces *second-order substitution* of all Σ-trees, i. e., substitution of trees for operation symbols, see [C]. We present a generalization of second-order substitution to the final coalgebras given by $\mathcal{T}(H)$.

In fact, in [AAMV, Mi$_1$] it is proved that the monad $\mathcal{T}(H)$ of Theorem 3.13 is characterized by an important universal property—it is the free *completely iterative monad* on H. This freeness of $\mathcal{T}(H)$ specializes to second-order substitution of Σ-trees, a fact we illustrate at the end of the current section.

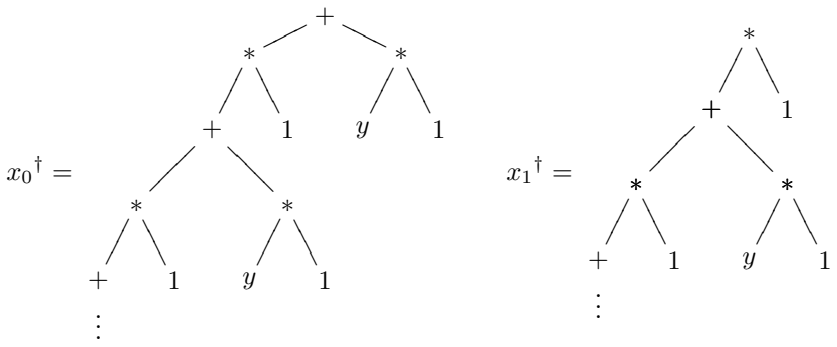
Here we shall quickly recall those results of [AAMV] which we will need in the current paper. For a well-motivated and more detailed exposition of the material presented here we refer the reader to one of [AAMV, Mi$_2$].

Example 4.1. We have seen in Section 3 that for a signature Σ, flat systems of formal equations have unique solutions whose components are Σ-trees over a set of parameters. But it is also possible to uniquely solve certain non-flat systems equations. More precisely, for a given signature Σ, consider a system of equations (3.1) where each right-hand side t_i, $i \in I$ is any Σ-tree from $T_\Sigma(X + Y)$ which is not just a single variable from X. Such systems are called *guarded*. Guardedness suffices to ensure the existence of a unique solution of the given system.

For example, let Σ consist of binary operations $+$ and $*$ and a constant 1. The following system of equations



with $X = \{ x_0, x_1 \}$ and $Y = \{ y \}$ is guarded. The solution is given by the following trees in $T_\Sigma Y$:



Unique solutions of guarded equations can be obtained more generally for every monad $\mathcal{T}(H)$ of Theorem 3.13 above. Before we state this result precisely we recall the notion of an ideal monad. It adds to an arbitrary monad S enough structure to be able to speak of guarded equations for S.

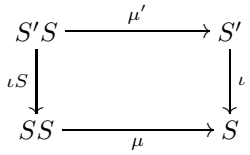
Definition 4.2. By an *ideal monad* we mean a six-tuple

$$(S, \eta, \mu, S', \iota, \mu')$$

consisting of a monad (S, η, μ), and a (right) *ideal* (S', μ'), which consists of a subfunctor $\iota : S' \hookrightarrow S$, i. e., a monomorphism ι in the functor category $[\mathcal{A}, \mathcal{A}]$, and a natural transformation $\mu' : S'S \longrightarrow S'$ such that the following two conditions hold:

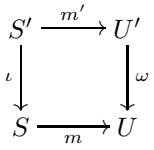
(i) $S = S' + Id$ with coproduct injections ι and η

(ii) μ restricts to μ' along ι, i.e., the square

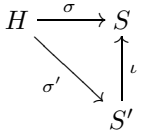


commutes.

An *ideal monad morphism* from an ideal monad $(S, \eta^S, \mu^S, S', \iota, \mu'^S)$ to an ideal monad $(U, \eta^U, \mu^U, U', \omega, \mu'^U)$ is a monad morphism $m : (S, \eta^S, \mu^S) \longrightarrow (U, \eta^U, \mu^U)$ which has a domain-codomain restriction to the ideals. That is, there exists a natural transformation $m' : S' \longrightarrow U'$ such that the square below commutes:



For any endofunctor H and ideal monad S, a natural transformation $\sigma : H \longrightarrow S$ is *ideal* if it factors through the ideal $\iota : S' \hookrightarrow S$ as follows:

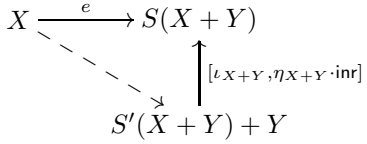


Example 4.3. Recall that the underlying functor of the monad T of Theorem 3.13 is a coproduct of HT and Id. Taking for ι the left-hand coproduct injection $\tau : HT \hookrightarrow T$ and for μ' the natural transformation $H\mu : HTT \longrightarrow HT$ we see that T is an ideal monad. Furthermore, since κ in (3.7) is $\tau \cdot H\eta$, $\kappa : H \longrightarrow T$ is an ideal natural transformation.

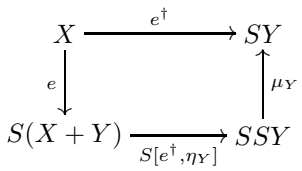
Definition 4.4. (i) For an ideal monad S on $\mathcal{A}$ an *equation morphism* is a morphism

$$e : X \longrightarrow S(X + Y).$$

It is called *guarded* if it factors as follows:



(ii) A *solution* of an equation morphism e is a morphism $e^\dagger : X \longrightarrow SY$ such that the following square



commutes.

(iii) An ideal monad is called *completely iterative* provided that any guarded equation morphism has a unique solution.

The first item of the following result is called the *Parametric Corecursion Theorem* in [Mo$_1$] and the *Solution Theorem* in [AAMV]. See also [Mi$_2$] for an extension of this result to all cias. The second item is the main result of [AAMV].

Theorem 4.5. *For any iteratable endofunctor H,*

(i) *the ideal monad $T = \mathfrak{T}(H)$ is completely iterative, and*

(ii) *T is free on H. More precisely, for all completely iterative monads S and ideal natural transformations $\sigma : H \longrightarrow S$ there exists a unique monad morphism $\overline{\sigma} : T \longrightarrow S$ such that $\overline{\sigma} \cdot \kappa^H = \sigma$:*

$$\begin{array}{ccc} H & \xrightarrow{\kappa^H} & T \\ & \searrow^{\forall \sigma} & \downarrow \exists! \overline{\sigma} \\ & & S\,. \end{array} \tag{4.1}$$

And the induced morphism $\overline{\sigma}$ is an ideal monad morphism.

In our work in the subsequent sections we shall often use the special case of Theorem 4.5 where the completely iterative monad S is $\mathcal{T}(K)$ for some iteratable endofunctor K. For that special case we need the following explicit description of the restriction of the monad morphism $\overline{\sigma}$ to the subfunctors $H\mathcal{T}(H)$ and $S' = K\mathcal{T}(K)$.

Lemma 4.6. *If H and K are iteratable endofunctors and $\sigma : H \longrightarrow \mathcal{T}(K)$ is an ideal transformation, i. e., $\sigma = \tau^K \cdot \sigma'$, then for the unique induced ideal monad morphism $\overline{\sigma}$ the following is a commutative diagram:*

$$\begin{array}{ccccc} H\mathcal{T}(H) & \xrightarrow{\sigma' * \overline{\sigma}} & K\mathcal{T}(K)\mathcal{T}(K) & \xrightarrow{K\mu^K} & K\mathcal{T}(K) \\ \downarrow{\scriptstyle \tau^H} & & & & \downarrow{\scriptstyle \tau^K} \\ \mathcal{T}(H) & & \xrightarrow[\overline{\sigma}]{} & & \mathcal{T}(K) \end{array}$$

Proof. Consider the diagram

$$\begin{array}{ccccc} H\mathcal{T}(H) & \xrightarrow{\sigma' * \overline{\sigma}} & K\mathcal{T}(K)\mathcal{T}(K) & \xrightarrow{K\mu^K} & K\mathcal{T}(K) \\ \downarrow{\scriptstyle \kappa^H * \mathcal{T}(H)} & & \downarrow{\scriptstyle \tau^K * \mathcal{T}(K)} & & \\ \mathcal{T}(H)\mathcal{T}(H) & \xrightarrow{\overline{\sigma} * \overline{\sigma}} & \mathcal{T}(K)\mathcal{T}(K) & \searrow^{\mu^K} & \downarrow{\scriptstyle \tau^K} \\ \downarrow{\scriptstyle \mu^H} & & & & \\ \mathcal{T}(H) & & \xrightarrow[\overline{\sigma}]{} & & \mathcal{T}(K) \end{array}$$

(with τ^H from the upper left to the lower left)

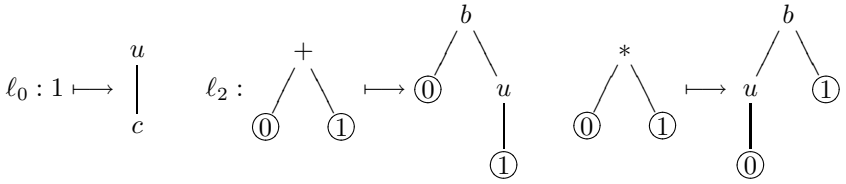
It clearly commutes. The left-hand and right-hand parts commute by Corollary 3.15. The upper part commutes by naturality and since $\overline{\sigma} \cdot \kappa^H = \sigma = \tau^K \cdot \sigma'$, and the lower one since $\overline{\sigma}$ is a monad morphism. □

For signature functors on Set, the freeness of $T = \mathcal{T}(H_\Sigma)$ specializes to *second-order substitution*, i. e., substitution of (finite or infinite) trees for operation symbols. Second-order substitution is a key point in the classical theory of recursive program schemes because the notion of an *uninterpreted solution* rests on it. We believe that the connection of second-order substitution and any notion of freeness is new.

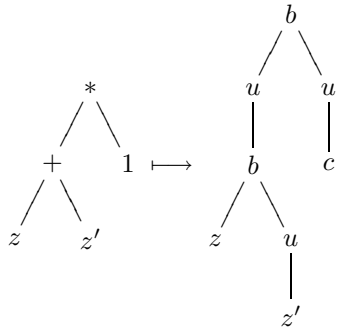
Example 4.7. Let Σ and Γ be signatures (considered as functors $\mathbb{N} \longrightarrow \mathsf{Set}$). Each symbol $\sigma \in \Sigma_n$ is considered as a flat tree in n variables. A second-order substitution gives an "implementation" to each such σ as a Γ-tree in the same n variables. We model this by a natural transformation $\ell : \Sigma \longrightarrow T_\Gamma \cdot J$, i. e., a family or maps $\ell_n : \Sigma_n \longrightarrow T_\Gamma\{0, \ldots n-1\}$, $n \in \mathbb{N}$. As we have seen in Section 2.3, this gives rise to a natural transformation $\lambda : H_\Sigma \longrightarrow T_\Gamma$. When infinite trees are involved there is usually the restriction to so-called *non-erasing* substitutions; i. e., all Σ-symbols are assigned to trees which are not just single node trees labelled by a variable. That means that λ is an ideal natural transformation. Thus, from Theorem 4.5 we get a monad morphism $\overline{\lambda} : T_\Sigma \longrightarrow T_\Gamma$. For any set X of variables its action is that of second-order substitution, i. e., $\overline{\lambda}_X$ replaces every Σ-symbol in a tree t from $T_\Sigma X$ by its implementation according to λ. More precisely, let $t = \sigma(t_1, \ldots, t_n)$ with $\sigma \in \Sigma_n$ and let $t'(x_1, \ldots x_n) \in T_\Gamma X$ be the implementation of σ, i. e., $\lambda_X(\sigma(\vec{x})) = t'(\vec{x})$. Then we have

$$\overline{\lambda}_X(t) = t'(\overline{\lambda}_X(t_1), \ldots, \overline{\lambda}_X(t_n)).$$

For example, suppose that Σ consists of two binary symbols $+$ and $*$ and a constant 1, and Γ consists of a binary symbol b, a unary one u and a constant c. Furthermore, let λ be given by $\ell : \Sigma \longrightarrow T_\Gamma \cdot J$ as follows:



and else ℓ_n is the unique map from the empty set. For the set $Z = \{ z, z' \}$, the second-order substitution morphism $\overline{\lambda}_Z$ acts for example as follows:



5. $\mathcal{T}(H)$ as Final Coalgebra among Monads

In this section we will state and prove some technical results which are essential for the proofs of our results on uninterpreted and interpreted program schemes. The culmination of the work comes in Corollary 5.5.

We would like to mention that the results and proofs in Section 5.1 are inspired by the work of Ghani et al. [GLM]. However, that paper does not work in the same category as we do. Our setting is slightly more general, and perhaps, conceptually slightly clearer. We do not believe that any of our subsequent new results in Sections 6 and 7 can be obtained by simply applying the work of [GLM].

We still assume that every functor H we consider is an iteratable endofunctor. Recall that for each object Y, TY is a final coalgebra for the functor $H(\_) + Y$ on $\mathcal{A}$.

We are going to prove a number of results that strengthen this. First, consider the functor category $[\mathcal{A}, \mathcal{A}]$. H may be regarded as a functor on this, by $F \longmapsto H \cdot F$. We also get a related functor $H \cdot \_ + Id$. For the functor T, the value of this functor at T is $H \cdot T + Id$. So the natural transformation $\alpha^H : T \longrightarrow HT + Id$ of Theorem 3.13(ii) may be regarded as a coalgebra structure for T. The proof of the following theorem is straightforward and therefore left to the reader.

Theorem 5.1. *(T, α) is a final coalgebra for $H \cdot \_ + Id$.*

5.1. T Gives a Final Coalgebra as a Monad.

We next consider the comma-category

$$H/\mathbf{Mon}(\mathcal{A}),$$

whose objects are given by pairs $(S, \sigma : H \longrightarrow S)$, where S is a monad on $\mathcal{A}$, and morphisms $h : (S_1, \sigma_1) \longrightarrow (S_2, \sigma_2)$ are monad morphisms $h : S_1 \longrightarrow S_2$ such that $h \cdot \sigma_1 = \sigma_2$. For example, one object of $H/\mathbf{Mon}(\mathcal{A})$ is (T, κ), where $\kappa = \tau \cdot H\eta$ is the canonical natural transformation of Theorem 3.13(iv). We show that H determines an endofunctor $\mathcal{H}$ on this category, and that (T, κ) is the underlying object of a final $\mathcal{H}$-coalgebra. We then extend this finality result by considering a subcategory of $H/\mathbf{Mon}(\mathcal{A})$. We slightly abuse notation and denote by $H/\mathbf{CIM}(\mathcal{A})$ the category whose objects are the pairs (S, σ), where S is *completely iterative* and σ is an *ideal* transformation; the morphisms in $H/\mathbf{CIM}(\mathcal{A})$ are given by the ideal monad morphisms, see Definitions 4.2 and 4.4. Again, (T, κ) is an object in this category, and we show that $\mathcal{H}$ restricts to an endofunctor on $H/\mathbf{CIM}(\mathcal{A})$, and that (T, κ) is once again a final coalgebra for $\mathcal{H}$.

Let us begin by defining $\mathcal{H}$ on objects of $H/\mathbf{Mon}(\mathcal{A})$ as the assignment

$$\mathcal{H} : (S, \sigma) \longmapsto (HS + Id, \mathsf{inl} \cdot H\eta)\,,$$

and for a morphism $h : (S_1, \sigma_1) \longrightarrow (S_2, \sigma_2)$ we let $\mathcal{H}(h) = Hh + Id$.

Furthermore, notice that for any object (S, σ) of $H/\mathbf{Mon}(\mathcal{A})$ there is a natural transformation

$$\xi_{(S,\sigma)} \equiv HS + Id \xrightarrow{[\mu \cdot \sigma S, \eta]} S$$

As it turns out, the $\xi_{(S,\sigma)}$ are the components of a natural transformation $\xi : \mathcal{H} \longrightarrow Id$ turning $\mathcal{H}$ into a well-copointed endofunctor on $H/\mathbf{Mon}(\mathcal{A})$.

Lemma 5.2.

(i) *$\mathcal{H}$ is an endofunctor of $H/\mathbf{Mon}(\mathcal{A})$;*
(ii) *$\xi : \mathcal{H} \longrightarrow Id$ is a natural transformation;*
(iii) *The functor $\mathcal{H}$ is well-copointed. That is, $\xi : \mathcal{H} \longrightarrow Id$ is a natural transformation with $\mathcal{H}\xi_{(S,\sigma)} = \xi_{\mathcal{H}(S,\sigma)}$ for all objects (S,σ) of $H/\mathbf{Mon}(\mathcal{A})$.*

Proof. (i) Given the object (S,σ) we define a natural transformation ν as

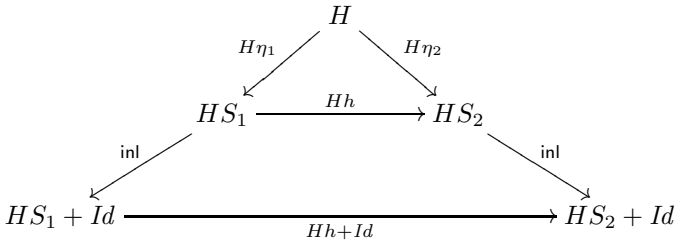
$$\begin{array}{c} (HS+Id)^2 = HS(HS+Id)+HS+Id \\ \Big\downarrow{\scriptstyle HS[\mu\cdot\sigma S,\eta]+HS+Id} \\ HSS+HS+Id \\ \Big\downarrow{\scriptstyle [H\mu,\mathsf{inl}]+Id} \\ HS+Id \end{array}$$

It is easy to check that $(HS+Id, \mathsf{inr}, \nu)$ is a monad, see [Mi$_1$], Lemma 3.4. Together with the natural transformation

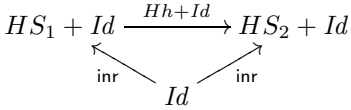
$$H \xrightarrow{H\eta} HS \xrightarrow{\mathsf{inl}} HS+Id$$

we obtain an object of $H/\mathbf{Mon}(\mathcal{A})$.

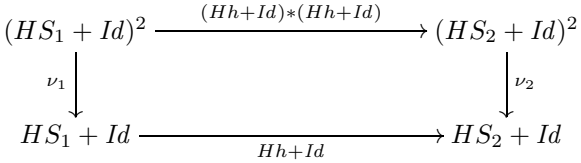
Now suppose that $h : (S_1,\sigma_1) \longrightarrow (S_2,\sigma_2)$ is a morphism in $H/\mathbf{Mon}(\mathcal{A})$. We establish below that $\mathcal{H}(h) = Hh + Id$ is a monad morphism. Together with the commutativity of the diagram



this establishes the action of $\mathcal{H}$ on morphisms. That $\mathcal{H}$ preserves identities and composition is obvious. Let us check then that $\mathcal{H}(h)$ is a monad morphism. The unit law is the commutativity of the triangle

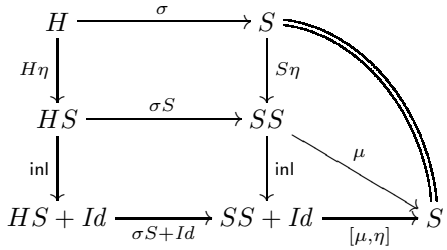


Thus, to complete the proof of (i) we must check that the following square commutes:

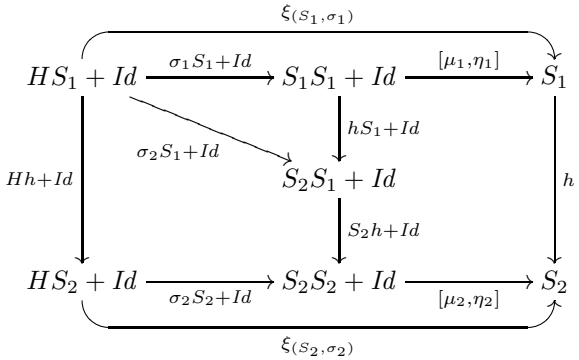


Expanding by using the definition of ν_i, $i = 1,2$, this is an essentially easy chase through some large diagrams using only the monad laws for the S_i as well as the fact that h is a morphism in $H/\mathbf{Mon}(\mathcal{A})$. For the sake of brevity we leave this straightforward task to the reader.

(ii) It is easy to prove that each component $\xi_{(S,\sigma)}$ is a monad morphism, see the proof of Theorem 3.1 in [Mi$_1$]. Moreover, by the commutativity of the following diagram, we obtain a morphism in $H/\mathbf{Mon}(\mathcal{A})$:



Finally, we check naturality of ξ. Suppose that $h : (S_1, \sigma_1) \longrightarrow (S_2, \sigma_2)$ is a morphism in $H/\mathbf{Mon}(\mathcal{A})$. Then, the following diagram commutes:



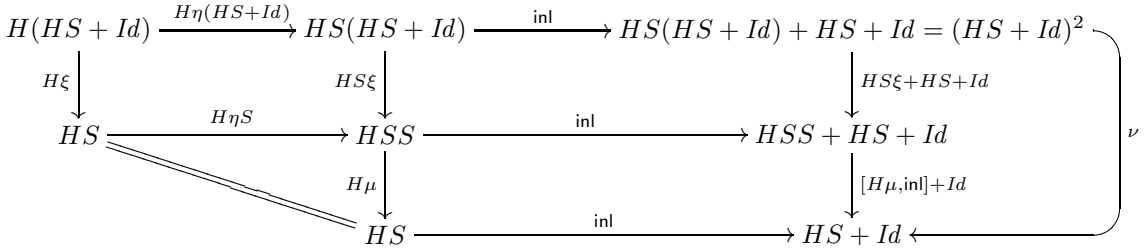
On the left we are using the naturality of σ_2, on the right and in the triangle that h is a morphism of $H/\mathbf{Mon}(\mathcal{A})$.

(iii) For any object (S, σ) we have by definition that

$$\xi_{\mathcal{H}(S,\sigma)} \equiv [\nu \cdot \mathsf{inl} \cdot H\eta(HS + Id), \mathsf{inr}] : H(HS + Id) + Id \longrightarrow HS + Id.$$

We show that this is the same as $H\xi_{(S,\sigma)} + Id$.

We consider the components of the left-hand coproduct separately. Equality on the right-hand component is obvious. For the left-hand one consider the following diagram:



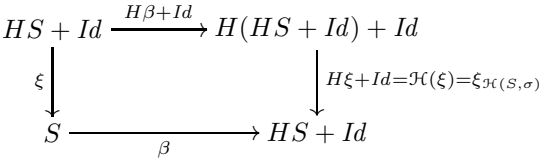
The upper and right outer edges compose to yield the left-hand component of $\xi_{\mathcal{H}(S,\sigma)}$ and the left and lower outer edges yield the left-hand component of $H\xi_{(S,\sigma)} + Id$. The desired commutativity of the outer shape follow since all inner parts of the diagram trivially commute. □

Lemma 5.3. *Let $((S, \sigma), \beta)$ be an $\mathcal{H}$-coalgebra with the structure $\beta : (S, \sigma) \longrightarrow \mathcal{H}(S, \sigma)$ in $H/\mathbf{Mon}(\mathcal{A})$. Then*

$$\xi_{(S,\sigma)} : \mathcal{H}(S, \sigma) \longrightarrow (S, \sigma)$$

is an $\mathcal{H}$-coalgebra homomorphism.

Proof. It is clear that $\mathcal{H}(S, \sigma) = (HS + Id, \mathsf{inl} \cdot H\eta)$ is an $\mathcal{H}$-coalgebra with the structure $H\beta + Id$, and that $\xi_{(S,\sigma)}$ is a morphism in $H/\mathbf{Mon}(\mathcal{A})$. From the naturality and well-copointedness of ξ, the following square



commutes; (see Lemma 5.2). □

Theorem 5.4. *$((T,\kappa),\alpha)$ is a final coalgebra for the functor $\mathcal{H}$ on $H/\mathbf{Mon}(\mathcal{A})$.*

Proof. Recall that the coalgebra structure $\alpha : T \longrightarrow HT + Id$ is given by the inverse of $[\tau,\eta] : HT + Id \longrightarrow T$. It is clearly a morphism in $H/\mathbf{Mon}(\mathcal{A})$. Indeed, recall from Corollary 3.15 that $\tau = \mu \cdot \kappa T$. Hence, $[\tau,\eta] = \xi_{(T,\kappa)}$ and so α being an inverse of a monad morphism is itself a monad morphism, and clearly we have $\alpha \cdot \kappa = \mathsf{inl} \cdot H\eta$.

Now suppose that $\beta : (S,\sigma) \longrightarrow \mathcal{H}(S,\sigma)$ is any $\mathcal{H}$-coalgebra. So the natural transformation $\beta : S \longrightarrow HS + Id$ is a monad morphism such that $\beta \cdot \sigma = \mathsf{inl} \cdot H\eta^S$. By Theorem 5.1, T is a final coalgebra on the level of endofunctors. Thus there exists a unique natural transformation $h : S \longrightarrow T$ such that the following square commutes:

$$\begin{array}{ccc} S & \xrightarrow{\beta} & HS + Id \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Hh+Id} \\ T & \xrightarrow[\alpha]{} & HT + Id \end{array} \tag{5.1}$$

Hence our only task is to show that h is a morphism in $H/\mathbf{Mon}(\mathcal{A})$. With an easy computation we establish the unit law:

$$\begin{aligned} h \cdot \eta^S &= \alpha^{-1} \cdot (Hh + Id) \cdot \beta \cdot \eta^S && \text{(by (5.1))} \\ &= [\tau,\eta] \cdot (Hh + Id) \cdot \mathsf{inr} && \text{(since } \alpha^{-1} = [\tau,\eta]\text{)} \\ &= [\tau,\eta] \cdot \mathsf{inr} \\ &= \eta && \text{(computation with inr)} \end{aligned}$$

and from this it follows that

$$\begin{aligned} h \cdot \sigma &= \alpha^{-1} \cdot (Hh + Id) \cdot \beta \cdot \sigma && \text{(by (5.1))} \\ &= [\tau,\eta] \cdot (Hh + Id) \cdot \mathsf{inl} \cdot H\eta^S && \text{(since } \alpha^{-1} = [\tau,\eta]\text{)} \\ &= \tau \cdot Hh \cdot H\eta^S && \text{(composition with inl)} \\ &= \tau \cdot H\eta && \text{(since } h \cdot \eta^S = \eta\text{)} \\ &= \kappa && \text{(by (3.7))}\,. \end{aligned}$$

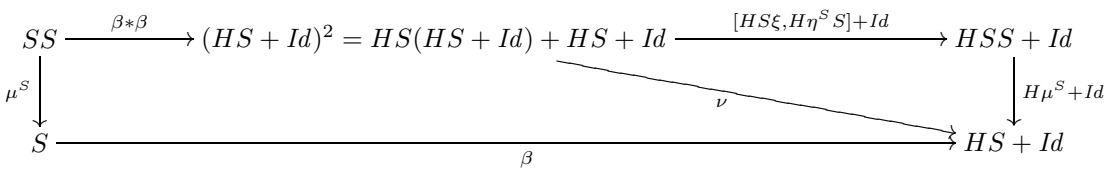
Finally, we check that the following square commutes:

$$\begin{array}{ccc} SS & \xrightarrow{\mu^S} & S \\ {\scriptstyle h*h}\downarrow & & \downarrow{\scriptstyle h} \\ TT & \xrightarrow[\mu]{} & T \end{array} \tag{5.2}$$

In order to do this we use that T is a final coalgebra for the functor $H \cdot \_ + Id : [\mathcal{A},\mathcal{A}] \longrightarrow [\mathcal{A},\mathcal{A}]$, and establish below that the arrows in square (5.2) are coalgebra homomorphisms. To do this, we need to first specify the coalgebra structures on the objects in (5.2). For S and T, we of course use β and α, respectively. For SS, we use the following coalgebra structure

$$SS \xrightarrow{\beta*\beta} (HS + Id)^2 = HS(HS + Id) + HS + Id \xrightarrow{[HS\xi, H\eta^S S] + Id} HSS + Id$$

It is easily established that $\mu^S : SS \longrightarrow S$ is a coalgebra homomorphism. Indeed, the following diagram commutes:



The left-hand part commutes since β is a monad morphism and the right-hand one is the definition of ν (use that $H\mu^S \cdot H\eta S = 1_{HS}$). Similarly, there is a coalgebra structure on TT such that $\mu : TT \longrightarrow T$ is a

coalgebra homomorphism. Finally, we show that $h * h : SS \longrightarrow TT$ is a coalgebra homomorphism. In fact, the following diagram commutes:

$$\begin{array}{ccccc}
SS & \xrightarrow{\beta*\beta} & (HS+Id)^2 = HS(HS+Id)+HS+Id & \xrightarrow{[HS\xi_{(S,\sigma)},H\eta^S S]+Id} & HSS+Id \\
\downarrow{\scriptstyle h*h} & & \downarrow{\scriptstyle (Hh+Id)^2=Hh(Hh+Id)+Hh+Id} & & \downarrow{\scriptstyle H(h*h)+Id} \\
TT & \xrightarrow[\alpha*\alpha]{} & (HT+Id)^2 = HT(HT+Id)+HT+Id & \xrightarrow[[HT\xi_{(T,\kappa)},H\eta T]+Id]{} & HTT+Id
\end{array}$$

The left-hand square commutes trivially. We consider the right-hand one componentwise: The right-hand component is obvious. For the middle one we remove H and obtain the following commutative square:

$$\begin{array}{ccc}
S & \xrightarrow{\eta^S S} & SS \\
\downarrow{\scriptstyle h} & & \downarrow{\scriptstyle h*h} \\
T & \xrightarrow[\eta T]{} & TT
\end{array}$$

Finally, for the left-hand component we remove H again to obtain

$$\begin{array}{ccc}
S(HS+Id) & \xrightarrow{S\xi_{(S,\sigma)}} & SS \\
\downarrow{\scriptstyle h*(Hh+Id)} & & \downarrow{\scriptstyle h*h} \\
T(HT+Id) & \xrightarrow[T\xi_{(T,\kappa)}]{} & TT
\end{array}$$

By naturality, it suffices to check that the square

$$\begin{array}{ccc}
HS+Id & \xrightarrow{\xi_{(S,\sigma)}} & S \\
\downarrow{\scriptstyle Hh+Id} & & \downarrow{\scriptstyle h} \\
HT+Id & \xrightarrow[\xi_{(T,\kappa)}]{} & T
\end{array}$$

commutes. Notice that we are not entitled to use naturality of ξ here, since we do not yet know that h is a monad morphism. Instead, we invoke the finality of T and show that all arrows in the above square are coalgebra homomorphisms for the functor $H \cdot \_ + Id$ on $[\mathcal{A}, \mathcal{A}]$. Indeed, since h is a coalgebra homomorphism, so is $Hh + Id$, and the other two morphisms are coalgebra homomorphisms by Lemma 5.3. This completes the proof. □

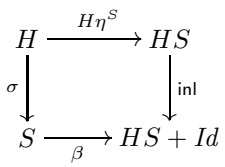
5.2. T **Gives a Final Coalgebra as a Completely Iterative Monad.** The next result is the main technical tool for our treatment of recursive program schemes in Section 6 below. Recall that we denote by $H/\mathbf{CIM}(\mathcal{A})$ the category whose objects are the pairs (S, σ), where S is completely iterative and σ is an ideal natural transformation; the morphisms in $H/\mathbf{CIM}(\mathcal{A})$ are given by the ideal monad morphisms. So $H/\mathbf{CIM}(\mathcal{A})$ is a subcategory of $H/\mathbf{Mon}(\mathcal{A})$. We show in Corollary 5.5 just below that $((T, \kappa), \alpha)$ is a final coalgebra for the same functor $\mathcal{H}$ that we have been working with. In the language of Theorem 5.4, the main point of the corollary is that if $\beta : S \longrightarrow HS + Id$ is an *ideal* monad morphism which is a coalgebra for $\mathcal{H}$ from some completely iterative monad, then the morphism into T may be taken to be an ideal monad morphism as well.

Corollary 5.5. *$\mathcal{H}$ restricts to an endofunctor on $H/\mathbf{CIM}(\mathcal{A})$, and $((T, \kappa), \alpha)$ is a final $\mathcal{H}$-coalgebra in $H/\mathbf{CIM}(\mathcal{A})$.*

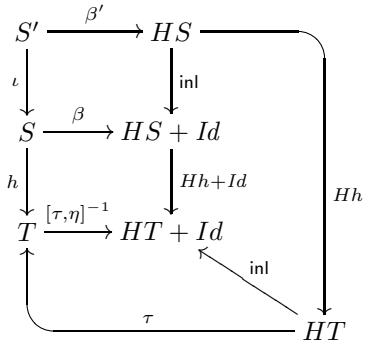
Proof. Lemma 3.5 of [Mi$_1$] gives a result propagating the complete iterative structure of monads: for any completely iterative monad S with ideal $\iota : S' \hookrightarrow S$, and any natural transformation $\sigma : H \longrightarrow S$ such that $\sigma = \iota \cdot \sigma'$ for some $\sigma' : H \longrightarrow S'$, the monad $HS + Id$ is completely iterative, too. Moreover, for any monad morphism h, $\mathcal{H}(h) = Hh + Id$ is an ideal monad morphism. Hence, $\mathcal{H}$ restricts to an endofunctor on $H/\mathbf{CIM}(\mathcal{A})$, which by abuse of notation we denote by $\mathcal{H}$ again.

Observe that (T, κ) lies in $H/\mathbf{CIM}(\mathcal{A})$ and that the coalgebra structure $\alpha = [\tau, \eta]^{-1}$ is an ideal monad morphism. Now suppose that $\beta : (S, \sigma) \longrightarrow (HS + Id, \mathsf{inl} \cdot H\eta^S)$ is an $\mathcal{H}$-coalgebra, i. e., β is an ideal monad

morphism between completely iterative monads such that the following square



commutes. By Theorem 5.4, we obtain a unique coalgebra homomorphism $h : (S, \sigma) \longrightarrow (T, \kappa)$ in $H/\mathbf{Mon}(\mathcal{A})$. Our only task is to check that h is an ideal monad morphism, so that it is a morphism in $H/\mathbf{CIM}(\mathcal{A})$. In fact, consider the following commutative diagram:



The middle square commutes since h is a coalgebra homomorphism, the upper one since β is an ideal monad morphism. The two other parts are obvious. Hence the outer shape commutes, proving that $Hh \cdot \beta' : S' \longrightarrow HT$ is a restriction of h to the ideal S' of S. □

6. Uninterpreted Recursive Program Schemes

In the classical treatment of recursive program schemes one gives an uninterpreted semantics to systems like (1.1) which are in *Greibach normal form*, i. e., every tree on the right-hand side of the system has as its head symbol a symbol from the given signature Σ. The semantics assigns to each of the new operation symbols a Σ-tree. These trees are obtained as the result of unfolding the recursive specification of the RPS. We illustrated this with an example in (1.7) in the Introduction.

We have seen in Section 2 that Σ-trees are the carrier of a final coalgebra of the signature functor H_Σ. It is the universal property of this final coalgebra which allows one to give a semantics to the given RPS. Using the technical tools developed in Section 5 we will now provide a conceptually easy and general way to give an uninterpreted semantics to recursive program schemes considered more abstractly as natural transformations, see our discussion in Section 2.3.

But before we do this, we need to say what a solution of an RPS should be. To do this we use the universal property of the monads $\mathcal{T}(H)$ as presented in Section 4. It gives an abstract version of second-order substitution.

Here are our central definitions, generalizing recursive program schemes from signatures to completely iterative monads.

Definition 6.1. Let V and H be endofunctors on $\mathcal{A}$. A *recursive program scheme* (or *RPS*, for short) is a natural transformation

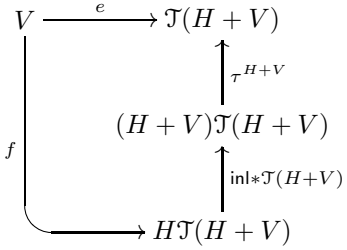
$$e : V \longrightarrow \mathcal{T}(H + V) .$$

We sometimes call V the *variables*, and H the *givens*.

The RPS e is called *guarded* if there exists a natural transformation

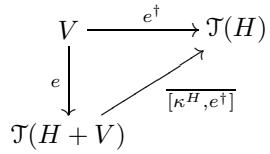
$$f : V \longrightarrow H\mathcal{T}(H + V)$$

such that the following diagram commutes:



(6.1)

A *solution* of e is an ideal natural transformation $e^\dagger : V \longrightarrow \mathcal{T}(H)$ such that the following triangle commutes:



(6.2)

Remark 6.2. Recall that $\overline{[\kappa^H, e^\dagger]}$ is the unique ideal monad morphism extending $\sigma = [\kappa^H, e^\dagger] : H + V \longrightarrow \mathcal{T}(H)$. Observe that therefore it is important to require that $e^\dagger$ be an ideal natural transformation since otherwise $\overline{\sigma}$ is not defined.

Remark 6.3.
(i) From Section 2.3, our definition is a generalization of the classical notion of RPS (to the category theoretic setting), and it extends the classical work as well by allowing infinite trees on the right-hand sides of equations.
(ii) Our notion of guardedness captures precisely the requirement that all right-hand sides of (2.3) have their root labelled by a symbol from the givens Σ. In the classical treatment of RPS this is precisely what is called Greibach normal form of an RPS, see [C].
(iii) Suppose that $H = H_\Sigma$ and $V = H_\Phi$ are signature functors of Set, and consider the recursive program scheme $e : H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$ as a set of formal equations as in (2.3). Then for any set X of syntactic variables the X-component $e^\dagger_X : H_\Phi X \longrightarrow T_\Sigma X$ of a solution assigns to any flat tree $(f, x_1, \ldots, x_n) = (f, \vec{x})$ from $H_\Phi X$ a Σ-tree over X. The commutativity of the triangle (6.2) gives the following property of solutions: apply to the right-hand side $t^f(\vec{x})$ of $f(\vec{x})$ in the given RPS the second-order substitution that replaces each operation symbol of Φ by its solution, and each operation symbol of Σ by itself—this is the action of $\overline{[\kappa^H, e^\dagger]}_X$. The resulting tree in $T_\Sigma X$ is the same tree as $e^\dagger_X(f, \vec{x})$.

Example 6.4. Let us now present two classical RPS as well as an example of RPS which is not captured with the classical setting.

(i) Recall from the Introduction the formal equations (1.1) and the ubiquitous (1.2) defining the factorial function. As explained in Example 2.8 these give rise to recursive program schemes. Thus, since both (1.1) and (1.2) are in Greibach normal form we obtain two guarded RPS's in the sense of Definition 6.1 above.
(ii) Sometimes one might wish to recursively define new operations from old ones where the new operations should satisfy certain extra properties automatically. We demonstrate this with an RPS recursively defining a new operation which is commutative. Suppose the signature Σ of givens consists of a ternary symbol F and a unary one G. Let us assume that we want to require that F satisfies the equation $F(x, y, z) = F(y, x, z)$ in any interpretation. This is modelled by the endofunctor $HX = X^3/\!\sim + X$ where $\sim$ is the smallest equivalence on X^3 with $(x, y, z) \sim (y, x, z)$. To be an H-algebra is equivalent to being an algebra A with a unary operation G_A and a ternary one F_A satisfying $F_A(x, y, z) = F_A(y, x, z)$. Suppose that one wants to define a commutative binary operation φ by the formal equation

$$\varphi(x, y) \approx F(x, y, \varphi(Gx, Gy)) \,. \tag{6.3}$$

To do it we express φ by the endofunctor V assigning to a set X the set $\{\, \{\, x, y \,\} \mid x, y \in X \,\}$ of unordered pairs of X. It is not difficult to see that the formal equation (6.3) gives rise to a guarded RPS $e : V \longrightarrow \mathcal{T}(H + V)$. In fact, to see the naturality use the description of the terminal coalgebra $\mathcal{T}(H + V)Y$ given in [AM], see Example 2.5(i). Notice that in the classical setting we are unable to demand that (the solution of) φ be a commutative operation at this stage: one would use general

facts to obtain a unique solution, and then one would need to devise a special argument to verify commutativity of that solution. Once again, our general theory insures that any solution of our RPS will be commutative.

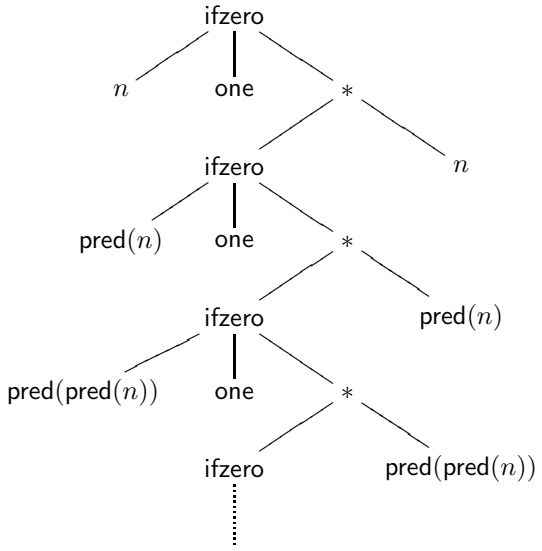
The main result of this section is the following theorem. Before we present its proof below let us illustrate the result with a few examples.

Theorem 6.5. *Every guarded recursive program scheme has a unique solution.*

Examples 6.6. We present here the solutions of the RPS's of Example 6.4.

(i) The unique solution of the RPS induced by the equations (1.1) is an ideal natural transformation $e^\dagger : H_\Phi \longrightarrow \mathcal{T}(H_\Sigma)$. Equivalently, we have a natural transformation $\Phi \longrightarrow T_\Sigma \cdot J$, see Section 2.3. That is, the solution $e^\dagger$ is essentially given by two Σ-trees (one for each of the operation symbols φ and ψ) over a singleton set, say $\{ x \}$. It follows from the proof of Theorem 6.5 that those Σ-trees are the ones given in (1.6), see Example 6.15 below.

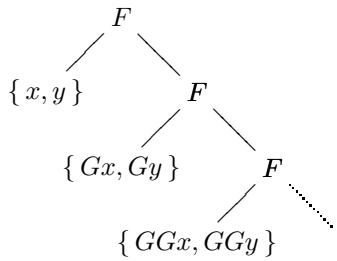
Similarly, the unique solution of the RPS induced by the equation (1.2) is essentially given by the Σ-tree over the set $\{ n \}$ below:



(6.4)

Notice that the nodes labelled by a term correspond to appropriately labelled finite subtrees.

(ii) We continue example 6.4(ii) and describe the uninterpreted solution of the guarded RPS e arising from the formal equation (6.3) defining a commutative operation. The components of $e^\dagger_X : VX \longrightarrow \mathcal{T}(H)X$ assign to an unordered pair $\{ x, y \}$ in VX the tree



where for every node labelled by F the order of the first two children cannot be distinguished; we indicate this with set-brackets in the picture above.

Remark 6.7. Notice that in the classical setting not every recursive program scheme which has a unique solution needs to be in Greibach normal form. For example, consider the system formed by the first equation in (1.1) and by the equation $\psi(x) \approx \varphi(\psi(x))$. This system gives rise to an unguarded RPS. Thus, Theorem 6.5 does not provide a solution of this RPS. However, the system is easily rewritten to an equivalent one in Greibach normal form which gives a guarded RPS that we can uniquely solve using our Theorem 6.5.

The rest of this section is devoted to the proof of Theorem 6.5. We illustrate each crucial step with the help of our examples. Before we turn to the proof of the main theorem we need to establish some preliminary lemmas.

Lemma 6.8. *Let H and K be endofunctors on $\mathcal{A}$. Suppose we have objects (S, σ) of $H/\mathbf{Mon}(\mathcal{A})$ and (R, ρ) of $K/\mathbf{Mon}(\mathcal{A})$. Let $n : H \longrightarrow K$ be a natural transformation and let $m : S \longrightarrow R$ be a monad morphism such that the following square*

$$\begin{array}{ccc} H & \xrightarrow{n} & K \\ {\scriptstyle\sigma}\downarrow & & \downarrow{\scriptstyle\rho} \\ S & \xrightarrow[m]{} & R \end{array}$$

commutes. Then $n * m + Id : HS + Id \longrightarrow KR + Id$ is a monad morphism such that the following square

$$\begin{array}{ccc} H & \xrightarrow{\quad n \quad} & K \\ {\scriptstyle \mathsf{inl}\cdot H\eta^S}\downarrow & & \downarrow{\scriptstyle \mathsf{inl}\cdot K\eta^R} \\ HS + Id & \xrightarrow[n*m+Id]{} & KR + Id \end{array}$$

commutes.

Proof. Naturality and the unit law are clear, and the preservation of the monad multiplication is a straightforward diagram chasing argument which we leave to the reader. □

Lemma 6.9. *Consider any guarded RPS e, see (6.1). There exists a unique (ideal) monad morphism*

$$\widehat{e} : \mathcal{T}(H + V) \longrightarrow \mathcal{T}(H + V)$$

such that the following triangle commutes:

$$\begin{array}{ccc} H + V & \xrightarrow{\kappa^{H+V}} & \mathcal{T}(H + V) \\ & {\scriptstyle [\kappa^{H+V}\cdot \mathsf{inl}, e]}\searrow & \downarrow{\scriptstyle \widehat{e}} \\ & & \mathcal{T}(H + V) \end{array}$$

There is also a unique (ideal) monad morphism

$$\overline{e} : \mathcal{T}(H + V) \longrightarrow H\mathcal{T}(H + V) + Id$$

such that the following diagram commutes:

$$\begin{array}{ccc} H + V & \xrightarrow{\kappa^{H+V}} & \mathcal{T}(H + V) \\ {\scriptstyle [H\eta^{H+V}, f]}\downarrow & & \downarrow{\scriptstyle \overline{e}} \\ H\mathcal{T}(H + V) & \xrightarrow[\mathsf{inl}]{} & H\mathcal{T}(H + V) + Id \end{array} \tag{6.5}$$

Proof. We get $\widehat{e}$ from Theorem 4.5. Indeed, it is easily checked that

$$[\kappa^{H+V} \cdot \mathsf{inl}, e] : H + V \longrightarrow \mathcal{T}(H + V)$$

is an ideal natural transformation using that e is guarded. As for $\overline{e}$, recall that H "embeds" into $\mathcal{T}(H + V)$ via the natural transformation

$$H \xrightarrow{\mathsf{inl}} H + V \xrightarrow{\kappa^{H+V}} \mathcal{T}(H + V)\,.$$

Since κ^{H+V} is an ideal natural transformation so is this one. Hence, $(\mathcal{T}(H + V), \kappa^{H+V} \cdot \mathsf{inl})$ is an object of $H/\mathbf{CIM}(\mathcal{A})$ and so it follows from Corollary 5.5 that $H\mathcal{T}(H + V) + Id$ carries the structure of a completely iterative monad. The natural transformation $\mathsf{inl} \cdot [H\eta^{H+V}, f] : H + V \longrightarrow H\mathcal{T}(H + V) + Id$, see (6.5), is obviously ideal. Thus, we obtain $\overline{e}$ as desired from another application of Theorem 4.5. □

Remark 6.10. In the leading example of a classical RPS for given signatures, the formation of the morphism $\overline{e}$ corresponds to the formation of a flat system of equations, where for every tree there is a recursion variable. More precisely, suppose we have signatures Σ and Φ, and an RPS as in (2.3) which is in Greibach normal form. The component of $\overline{e}$ at some set Z of syntactic variables can be seen as a set of formal equations. Here is a description of $\overline{e}_Z$: For every tree $t \in T_{\Sigma+\Phi}Z$, we have a recursion variable $\underline{t}$. And for each recursion variable we have one formal equation:

$$\begin{array}{lcll} \underline{t} & \approx & x & \text{if } t \text{ is a single node tree with root label } x \in Z, \text{ or} \\ \underline{t} & \approx & \sigma(\underline{t_1}, \ldots, \underline{t_n}) & \text{for some } n \in \mathbb{N} \text{ and some } \sigma \in \Sigma_n \text{ otherwise,} \end{array}$$

where the tree $s = \sigma(t_1, \ldots, t_n)$ is the result of the following second-order substitution applied to t: every symbol of Φ is substituted by its right-hand side in the given RPS, and every symbol of Σ by itself. Since the given RPS is guarded the head symbol of s is a symbol of Σ for all trees t.

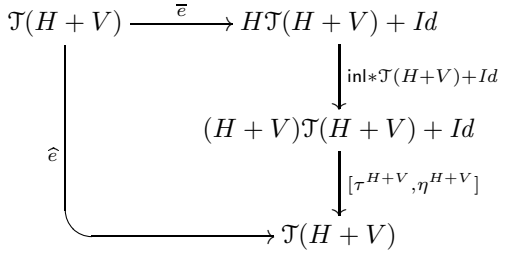
Observe that forming the right-hand sides of this system corresponds to the application of one step of Kleene's computation rule, see [G].

Example 6.11. For the guarded RPS of (1.1) the flat system obtained from $\overline{e}$ for $Z = \{ x \}$ includes the equations of the system (1.7) from the Introduction.

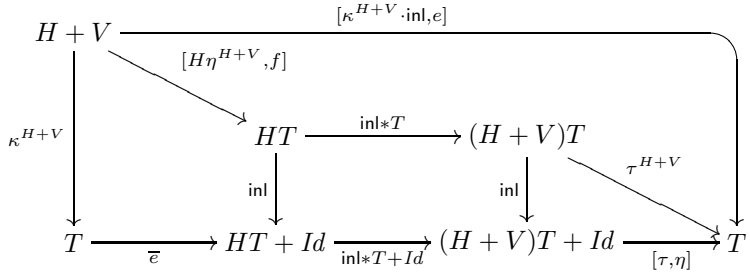
We also give a fragment of the flat system obtained as the extension of the RPS (1.2), see also Example 6.4(i) . Here the set of syntactic variables is $Z = \{n\}$ and the formal equations described by $\overline{e}_Z$ include the following ones:

$$\begin{aligned}
\underline{f(n)} &\approx \mathsf{ifzero}(\underline{n}, \underline{\mathsf{one}}, \underline{f(\mathsf{pred}(n)) * n}) \\
\underline{n} &\approx n \\
\underline{\mathsf{one}} &\approx \mathsf{one} \\
\underline{f(\mathsf{pred}(n)) * n} &\approx \underline{\mathsf{ifzero}(\mathsf{pred}(n), \mathsf{one}, f(\mathsf{pred}(\mathsf{pred}(n)))) * \underline{n}} \\
\underline{f(\mathsf{pred}(n))} &\approx \mathsf{ifzero}(\underline{\mathsf{pred}(n)}, \underline{\mathsf{one}}, \underline{f(\mathsf{pred}(\mathsf{pred}(n)) * \mathsf{pred}(n))}) \\
&\vdots
\end{aligned} \tag{6.6}$$

Lemma 6.12. *The following diagram commutes:*



Proof. By the first part of Lemma 6.9, it suffices to show that the composite in the above diagram is an ideal monad morphism extending $[\kappa^{H+V} \cdot \mathsf{inl}, e]$. For the extension property, consider the following commutative diagram, where we write T for $\mathcal{T}(H+V)$:



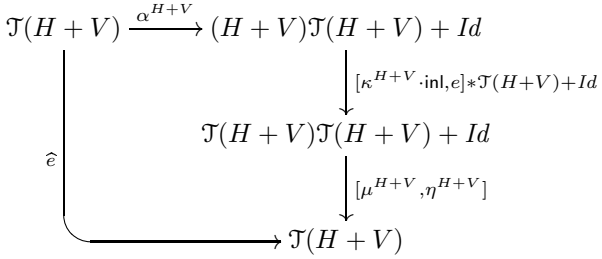
The left-hand part commutes by Lemma 6.9. For the left-hand component of the upper part notice that

$$\tau^{H+V} \cdot (\mathsf{inl} * T) \cdot H\eta^{H+V} = \tau^{H+V} \cdot (H+V)\eta^{H+V} \cdot \mathsf{inl} = \kappa^{H+V} \cdot \mathsf{inl}\,.$$

The right-hand component of this part commutes since e is guarded, see (6.1), and the remaining parts are trivial.

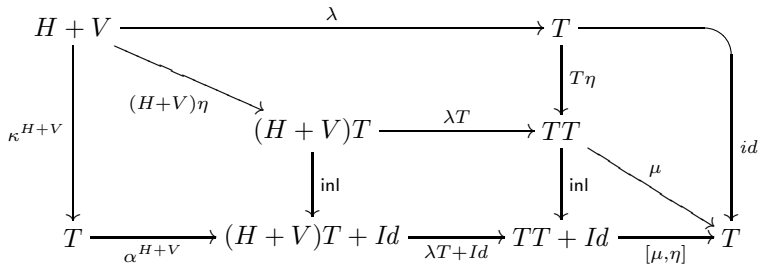
We show that all parts of the lower edge in the above diagram are monad morphisms. For $\overline{e}$, see Lemma 6.9. For $\mathsf{inl} * T + Id$, apply Lemma 6.8 to $n = \mathsf{inl}$ and $m = 1_T$. And the for the last part, $[\tau, \eta] = [\mu \cdot \kappa T, \eta]$, notice that it is the component at $(\mathcal{T}(H+V), \kappa^{H+V})$ of the natural transformation ξ of Lemma 5.2 applied to $(H+V)/\mathbf{Mon}(\mathcal{A})$. □

Lemma 6.13. *The following diagram*



commutes.

Proof. By the first part of Lemma 6.9, it suffices to show that the composite in the above diagram is an ideal monad morphism extending $[\kappa^{H+V} \cdot \mathsf{inl}, e]$. Let us write λ for $[\kappa^{H+V} \cdot \mathsf{inl}, e]$ and T for $\mathcal{T}(H+V)$. Now consider the following commutative diagram



which establishes the extension part. To see that the morphism of the lower edge is a monad morphism, recall from Theorem 5.4 that α^{H+V} is the structure map of a final coalgebra of a functor on $(H+V)/\mathbf{Mon}(\mathcal{A})$, whence a monad morphism, and $[\mu \cdot \lambda T, \eta]$ is the component at (T, λ) of the natural transformation ξ, see Lemma 5.2. □

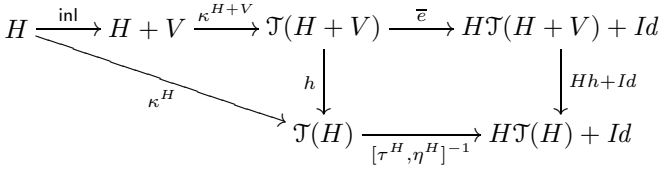
Proof of Theorem 6.5. Consider $\overline{e}$ from Lemma 6.9. It is a coalgebra structure for the functor

$$\mathcal{H} : H/\mathbf{CIM}(\mathcal{A}) \longrightarrow H/\mathbf{CIM}(\mathcal{A})$$

In fact, $\overline{e}$ is a morphism in $H/\mathbf{CIM}(\mathcal{A})$; it is an ideal monad morphism and by (6.5) we have

$$\overline{e} \cdot \kappa^{H+V} \cdot \mathsf{inl} = \mathsf{inl} \cdot [H\eta^{H+V}, f] \cdot \mathsf{inl} = \mathsf{inl} \cdot H\eta^{H+V}\,. \tag{6.7}$$

Now apply Corollary 5.5 to obtain a unique $\mathcal{H}$-coalgebra homomorphism from the above coalgebra $\overline{e}$ to the final one. In more detail, we obtain a unique ideal monad morphism $h : \mathcal{T}(H+V) \longrightarrow \mathcal{T}(H)$ such that the following diagram commutes:



(6.8)

Now let

$$e^\dagger \equiv V \xrightarrow{\mathsf{inr}} H+V \xrightarrow{\kappa^{H+V}} \mathcal{T}(H+V) \xrightarrow{h} \mathcal{T}(H)\,. \tag{6.9}$$

We shall prove that $e^\dagger$ uniquely solves e.

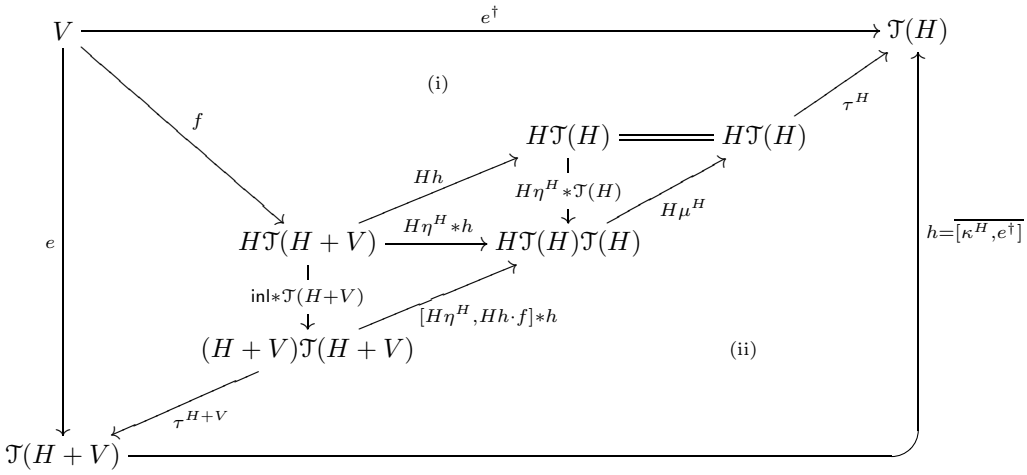
(a) $e^\dagger$ is a solution of e. Since h is an ideal monad morphism and κ^{H+V} is an ideal natural transformation, we see that $e^\dagger$ is an ideal natural transformation. Next observe that by definition we have

$$h = \overline{[\kappa^H, e^\dagger]},$$

and we also get

$$\begin{aligned} e^\dagger &= h \cdot \kappa^{H+V} \cdot \mathsf{inr} && \text{(by (6.9))} \\ &= [\tau^H, \eta^H] \cdot (Hh + Id) \cdot \overline{e} \cdot \kappa^{H+V} \cdot \mathsf{inr} && \text{(by (6.8))} \\ &= [\tau^H, \eta^H] \cdot (Hh + Id) \cdot \mathsf{inl} \cdot f && \text{(by (6.5))} \\ &= \tau^H \cdot Hh \cdot f. \end{aligned} \tag{6.10}$$

Now we can conclude that the following diagram commutes:



(6.11)

Indeed, part (i) commutes by (6.10). For part (ii) observe first that from (3.7) and (6.10) we get the equation

$$[\kappa^H, e^\dagger] \equiv H + V \xrightarrow{[H\eta^H, Hh \cdot f]} H\mathcal{T}(H) \xrightarrow{\tau^H} \mathcal{T}(H)\,. \tag{6.12}$$

Now apply Lemma 4.6 to $H+V$ and H and $\sigma = [\kappa^H, e^\dagger]$ using that (6.12) induces the ideal monad morphism $\overline{[\kappa^H, e^\dagger]} = h$. The other parts of (6.11) are obvious.

(b) Uniqueness of solutions. Suppose that $s : V \longrightarrow \mathcal{T}(H)$ is a solution of e. Since solutions are ideal natural transformations by definition, there exists a natural transformation $s' : V \longrightarrow H\mathcal{T}(H)$ such that $s = \tau^H \cdot s'$. We shall show below that the ideal monad morphism h from (6.8) is equal to

$$\overline{[\kappa^H, s]} : \mathcal{T}(H+V) \longrightarrow \mathcal{T}(H) \tag{6.13}$$

using coinduction, i. e., we show that $\overline{[\kappa^H, s]}$ is a coalgebra homomorphism. Then, since $(\mathcal{T}(H), \kappa^H)$ is a final $\mathcal{H}$-coalgebra, we can conclude that $h = \overline{[\kappa^H, s]}$ and therefore

$$e^\dagger = \overline{[\kappa^H, e^\dagger]} \cdot e = h \cdot e = \overline{[\kappa^H, s]} \cdot e = s,$$

where the last equality holds since s is a solution of e.

For the coinduction argument, replace h in Diagram (6.8) by $\overline{[\kappa^H, s]}$ and check that the modified diagram commutes. In fact, for the left-hand triangle we obtain:

$$\overline{[\kappa^H, s]} \cdot \kappa^{H+V} \cdot \mathsf{inl} = [\kappa^H, s] \cdot \mathsf{inl} = \kappa^H\,. \tag{6.14}$$

To verify the modified version of the right-hand square of (6.8), use the freeness of the completely iterative monad $\mathcal{T}(H+V)$. Thus, it is sufficient that this diagram of ideal monad morphisms commutes when precomposed with the universal arrow κ^{H+V}. Furthermore we consider the components of the coproduct $H+V$ separately. Let us write x as a short notation for $\overline{[\kappa^H, s]}$. Then, for the left-hand coproduct component we obtain the following equations:

$$\begin{aligned}
&[\tau^H, \eta^H] \cdot (Hx + Id) \cdot \overline{e} \cdot \kappa^{H+V} \cdot \mathsf{inl} \\
&\quad = [\tau^H, \eta^H] \cdot (Hx + Id) \cdot \mathsf{inl} \cdot H\eta^{H+V} && \text{(see (6.7))} \\
&\quad = \tau^H \cdot Hx \cdot H\eta^{H+V} \\
&\quad = \tau^H \cdot H\eta^H && \text{(since } x \cdot \eta^{H+V} = \eta^H\text{)} \\
&\quad = \kappa^H && \text{(see (3.7))} \\
&\quad = x \cdot \kappa^{H+V} \cdot \mathsf{inl} && \text{(see (6.14))}
\end{aligned}$$

In order to prove the right-hand component commutative we use a diagram similar to Diagram (6.11). Just replace in (6.11) $Hh \cdot f$ by s', all other occurrences of h by x, and $e^\dagger$ by s. We prove that part (i) in this modified diagram commutes. In fact, this follows from the fact that all other parts and the outward square commute: the outward square commutes since s is a solution of e; part (ii) in the modified diagram commutes by Lemma 4.6 again, and all other parts are clear. Thus, we proved that the following equation holds:

$$s = \tau^H \cdot Hx \cdot f. \tag{6.15}$$

From this we obtain the equations

$$\begin{aligned}
&[\tau^H, \eta^H] \cdot (Hx + Id) \cdot \overline{e} \cdot \kappa^{H+V} \cdot \mathsf{inr} \\
&\quad = \tau^H \cdot Hx \cdot f && \text{(similar as in (6.10))} \\
&\quad = s && \text{(by (6.15))} \\
&\quad = x \cdot \kappa^{H+V} \cdot \mathsf{inr} && \text{(since } x = \overline{[\kappa^H, s]}\text{)}\,.
\end{aligned}$$

This establishes that h from (6.9) is equal to (6.13). □

Remark 6.14. Recall that the formation of $\overline{e}$ corresponds, in the leading example of an RPS for given signatures, to the formation of a flat system of equations, see Remark 6.10. Now the map h_Z of (6.8) assigns to every variable $\underline{t} \in T_{\Sigma+\Phi}Z$ of the flat system the Σ-tree given by unfolding the recursive specification given by this flat system, i. e., h_Z is the unique solution of the flat equation morphism $\overline{e}_Z$ in the cia $T_\Sigma Z$.

Example 6.15. In equations (1.1) in the beginning of this paper, we introduced a guarded RPS as they are classically presented. It induces an RPS (in our sense), as discussed in Example 6.4(i). The unique solution of the flat equation morphism $\overline{e}_Z$ corresponding to the flat system described in Example 6.11 is $h_Z : T_{\Sigma+\Phi}Z \longrightarrow T_\Sigma Z$ (see also (1.7) from the Introduction). The definition of $e^\dagger$ in (6.9) means that we only consider the solution for the recursion variables $\underline{\varphi(x)}$ and $\underline{\psi(x)}$ in that system. These solutions are precisely the Σ-trees described in Example 6.6(i).

Similarly, for the guarded RPS induced by (1.2) the solution is obtained by considering only the unique solution for the variable $\underline{f(x)}$ of the flat system (6.6). Clearly, this yields the desired tree (6.4).

7. Interpreted Recursive Program Schemes

We have seen in the previous section that for every guarded recursive program scheme we can find a unique uninterpreted solution. In practice, however, one is more interested in finding *interpreted* solutions. In the classical treatment of recursive program schemes, this means that a recursive program scheme defining new operation symbols of a signature Φ from given ones in a signature Σ comes together with some Σ-algebra A. An interpreted solution of the recursive program scheme in question is, then, an operation on A for each operation symbol in Φ such that the formal equations of the RPS become valid identities in A.

Of course, in general an algebra A will not admit interpreted solutions. We shall prove in this section that any Elgot algebra $(A, a, (\_)^*)$ as defined in Section 3 admits an interpreted solution of any guarded recursive program scheme. Moreover, if A is a completely iterative algebra, interpreted solutions are unique. We also define the notion of a *standard* interpreted solution and prove that uninterpreted solutions and standard interpreted ones are consistent with one another as expected. This is a fundamental result for algebraic semantics.

We turn to applications after proving our main results. In Subsection 7.1 we study the computation tree semantics of RPS's arising from the computation tree Elgot algebras of Section 3.3. Then, in Subsection 7.2 we prove that in the category CPO our interpreted program scheme solutions agree with the usual denotational semantics obtained by computing least fixed points. Similarly, we show in Subsection 7.3 for the category of CMS that our solutions are the same as the ones computed using Banach's Fixed Point Theorem. Furthermore, we present new examples of recursive program scheme solutions pertaining to fractal self-similarity. We are not aware of any previous work connecting recursion to implicitly defined sets.

Definition 7.1. Let $(A, a, (\_)^*)$ be an Elgot algebra for H and let $e : V \longrightarrow \mathcal{T}(H + V)$ be an RPS. An *interpreted solution* of e in A is a structure of a V-algebra $e^\ddagger_A : VA \longrightarrow A$, such that

(i) the $(H + V)$-algebra $[a, e^\ddagger_A] : (H + V)A \longrightarrow A$ is the structure morphism of an Elgot algebra $(A, [a, e^\ddagger_A], (\_)^+)$ for $H + V$; and
(ii) the triangle below

$$\begin{array}{ccc} VA & \xrightarrow{\;e^\ddagger_A\;} & A \\ {\scriptstyle e_A}\downarrow & \nearrow_{[a,e^\ddagger_A]} & \\ \mathcal{T}(H+V)A & & \end{array} \tag{7.1}$$

commutes, where the diagonal arrow denotes the evaluation morphism associated to the Elgot algebra in (i), see Theorem 3.14.

Remark 7.2.

(i) The subscript A in $e^\dagger_A$ is only present to remind us of the codomain A. That is, $e^\dagger_A$ is not a component of any natural transformation.

(ii) Recall from Corollary 3.15 that the triangle

$$\begin{array}{ccc} (H+V)A & \xrightarrow{\kappa_A^{H+V}} & \mathcal{T}(H+V)A \\ & \searrow^{[a,e^\dagger_A]} & \downarrow{\scriptstyle \widetilde{[a,e^\dagger_A]}} \\ & & A \end{array} \tag{7.2}$$

commutes. It follows that for an interpreted solution $e^\dagger_A$, we have

$$a = \widetilde{[a, e^\dagger_A]} \cdot \kappa_A^{H+V} \cdot \mathsf{inl}, \tag{7.3}$$

(iii) In our leading example where $H = H_\Sigma$ and $V = H_\Phi$ are signature functors on Set, the commutativity of (7.1) states precisely that an interpreted solution provides operations on A which turn the formal equations of the given recursive program scheme into actual identities. More precisely, suppose that e is a recursive program scheme given by formal equations (2.3). The interpreted solution $e^\dagger_A$ gives for each n-ary operation symbol f of the signature Φ an operation $f_A : A^n \longrightarrow A$. And the commutativity of (7.1) gives the following property of f_A: take for any $\vec{a} \in A^n$ the right-hand side $t^f(\vec{a})$ in the given recursive program scheme, then evaluate $t^f(\vec{a})$ in A using the given operations for Σ and the ones provided by $e^\dagger_A$ for Φ on A—this is the action of $\widetilde{[a, e^\dagger_A]}$. The resulting element of A is the same as $f_A(\vec{a})$.

(iv) The requirement that $[a, e^\dagger_A]$ be the structure morphism of an Elgot algebra may seem odd at first. However, we need to assume this in order to be able to use $\widetilde{[a, e^\dagger_A]}$ in (7.1). Furthermore, the requirement has a clear practical advantage: operations defined recursively by means of an interpreted solution of an RPS may be used in subsequent recursive definitions. For example, for the signature functors on Set as in (iii) above the Elgot algebra with structure map $[a, e^\dagger_A]$ has operations for all operation symbols of $\Sigma + \Phi$. Thus, it can be used as an interpretation of givens for any further recursive program scheme with signature $\Sigma + \Phi$ of givens.

Theorem 7.3. *Let $(A, a, (\_)^*)$ be an Elgot algebra for H and let $e : V \longrightarrow \mathcal{T}(H + V)$ be a guarded RPS. Then the following hold:*

(i) *there exists an interpreted solution $e^\dagger_A$ of e in A,*

(ii) *if A is a completely iterative algebra, then $e^\dagger_A$ is the unique interpreted solution of e in A.*

We will present the proof after a technical lemma. It follows from the proof of Theorem 7.3 that uninterpreted solutions correspond to certain interpreted ones in a canonical way. We shall make this precise at the end of this subsection, and prove what could be called "Fundamental Theorem of Algebraic Semantics".

Lemma 7.4. *Let $(A, a, (\_)^*)$ be an Elgot algebra, let e be a guarded RPS, and let $\widehat{e}$ be as in Lemma 6.9. Then the following are in one-to-one correspondence:*

(i) *the interpreted solutions $e^\dagger_A$ of e in A.*

(ii) *the evaluation morphisms $\beta : \mathcal{T}(H + V)A \longrightarrow A$ such that the two diagrams*

$$\begin{array}{ccccc} HA & \xrightarrow{\mathsf{inl}_A} & (H+V)A & \xrightarrow{\kappa_A^{H+V}} & \mathcal{T}(H+V)A \\ & \searrow^{a} & & & \downarrow{\scriptstyle \beta} \\ & & & & A \end{array} \tag{7.4}$$

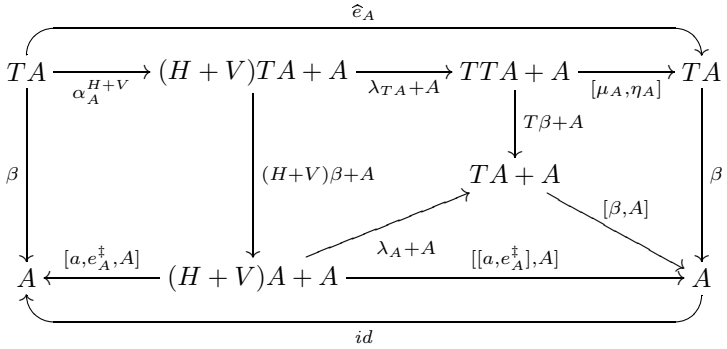
$$\begin{array}{ccc} \mathcal{T}(H+V)A & \xrightarrow{\widehat{e}_A} & \mathcal{T}(H+V)A \\ & \searrow^{\beta} & \downarrow{\scriptstyle \beta} \\ & & A \end{array} \tag{7.5}$$

commute.

In more detail, if $e^\dagger_A$ is an interpreted solution of e in A, then the evaluation morphism $\beta = \widetilde{[a, e^\dagger_A]}$ has the properties of (ii). And if β makes the diagrams in (ii) commute, then $\beta \cdot \kappa_A^{H+V} \cdot \mathsf{inr}$ is an interpreted solution to e in A. Finally, these two operations are mutually inverse.

Proof. Let us write T as a short notation for $\mathcal{T}(H+V)$ and λ for $[\kappa^{H+V} \cdot \mathsf{inl}, e] : H + V \longrightarrow T$.

(i) $\Rightarrow$ (ii): As in our statement, take the evaluation morphism $\beta = \widetilde{[a, e^\dagger_A]}$. Then (7.4) is (7.3). For (7.5), consider the following commutative diagram



The upper part commutes by Lemma 6.13, the right-hand part commutes since β is an Eilenberg-Moore algebra structure, and the left-hand part commutes since β is given as the solution of α_A^{H+V} in the Elgot algebra $(A, [a, e^\dagger_A], (\_)^+)$, see Theorem 3.13. For the middle part simply use the naturality of λ. The lowest part is obvious. Finally, for the commutativity of the lower triangle it suffices to show that the equation

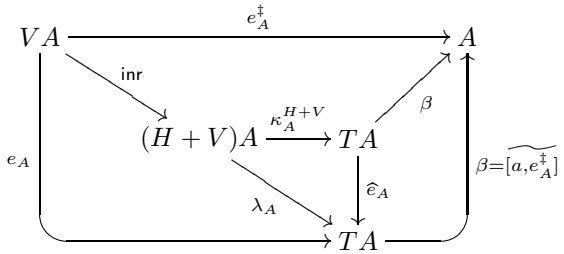
$$\beta \cdot \lambda_A = [a, e^\dagger_A]$$

holds. We consider the components separately: the left-hand one is (7.3), and the right-hand one is the triangle (7.1). Thus, the outer shape of the above diagram commutes, viz. the desired triangle (7.5).

(ii) $\Rightarrow$ (i): Define

$$e^\dagger_A \equiv VA \xrightarrow{\mathsf{inr}} (H+V)A \xrightarrow{\kappa_A^{H+V}} \mathcal{T}(H+V)A \xrightarrow{\beta} A\,. \tag{7.6}$$

It follows from the commutativity of (7.4) and Theorem 3.14 that $[a, e^\dagger_A] = \beta \cdot \kappa_A^{H+V}$ is the structure map of an Elgot algebra for $H + V$ on A so that the equation $\beta = \widetilde{[a, e^\dagger_A]}$ holds. For Diagram (7.1) consider the following commutative diagram:



The inner triangle commutes due to the definition of $\widehat{e}$, see Lemma 6.9, the right-hand one due to (7.5), and the other two parts are clear. Thus, the outer shape commutes, so we have (7.1).

Finally, we check that the operations of going from interpreted solutions $e^\dagger_A$ to evaluation morphisms β are mutually inverse. In fact, we have

$$\widetilde{[a, e^\dagger_A]} \cdot \kappa_A^{H+V} \cdot \mathsf{inr} = e^\dagger_A$$

by the commutativity of the right-hand component of Diagram (7.2). And for the interpreted solution $e^\dagger_A$ defined by (7.6), we have already seen above that the equation $\beta = \widetilde{[a, e^\dagger_A]}$ holds. This completes the proof. □

Remark 7.5. Lemma 7.4 above states that interpreted solutions correspond to suitable evaluation morphisms β such that the diagrams (7.4) and (7.5) commute. In particular, if $H = H_\Sigma$ and $V = H_\Phi$ are signature functors on Set, the existence of β means that all trees over the signature $\Sigma + \Phi$ can be evaluated in A. The commutativity of (7.4) states that for trees in $T_{\Sigma+\Phi}A$ operation symbols of Σ are interpreted according to the given Σ-algebra structure a, while the commutativity of (7.5) states that the operation symbols of Φ are interpreted by operations that satisfy the equations given by the recursive program scheme e.

Proof of Theorem 7.3. (i) Given an Elgot algebra $(A, a, (\_)^*)$ and a guarded recursive program scheme $e : V \longrightarrow \mathcal{T}(H+V)$ consider $\overline{e} : \mathcal{T}(H+V) \longrightarrow H \cdot \mathcal{T}(H+V) + Id$ from Lemma 6.9. Its component at A yields a flat equation morphism

$$g \equiv \mathcal{T}(H+V)A \longrightarrow H\mathcal{T}(H+V)A + A\,, \tag{7.7}$$

with respect to $(A, a, (\_)^*)$ and we take its solution

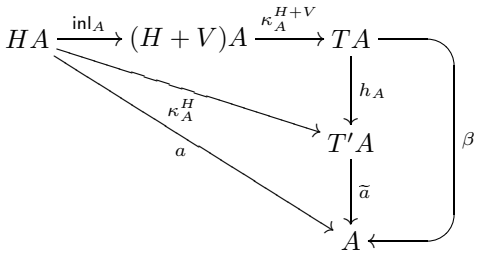
$$\beta \equiv \mathcal{T}(H+V)A \xrightarrow{g^*} A \tag{7.8}$$

By Lemma 7.4, it suffices to prove that β is an evaluation morphism such that the diagrams (7.4) and (7.5) commute. We first check that β is the structure of an Eilenberg-Moore algebra for $\mathcal{T}(H+V)$, hence an evaluation morphism. To this end we first establish the equation

$$\beta = \widetilde{a} \cdot h_A\,, \tag{7.9}$$

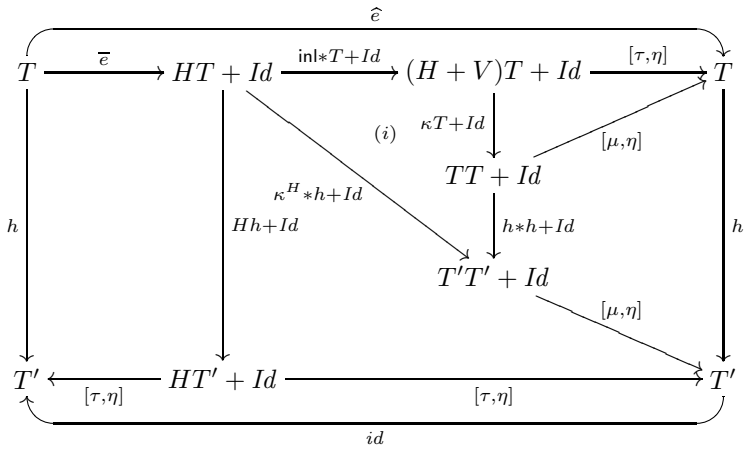
where $h : \mathcal{T}(H+V) \longrightarrow \mathcal{T}(H)$ is the monad morphism that we obtain from the recursive program scheme e. (See (6.8). It is also useful to recall that $h = \overline{[\kappa^H, e^\dagger]}$.) Recall that $\widetilde{a} = (\alpha_A)^*$, see Theorem 3.13, and that h_A is a homomorphism of $(H(\_) + A)$-coalgebras, see Diagram (6.8). Thus, by the Functoriality of $(\_)^*$, we obtain $g^* = (\alpha_A)^* \cdot h_A$, i. e., the desired equation (7.9). Now since $h : \mathcal{T}(H+V) \longrightarrow \mathcal{T}(H)$ is a monad morphism, and $\widetilde{a}$ is the structure morphism of an Eilenberg-Moore algebra for $\mathcal{T}(H)$, $\beta = \widetilde{a} \cdot h_A$ is an Eilenberg-Moore algebra for $\mathcal{T}(H+V)$. In fact, this follows from a general fact from category theory, see e. g. Proposition 4.5.9 in [Bo].

We check that the diagrams (7.4) and (7.5) commute. Let us use T as a short notation for $\mathcal{T}(H+V)$ and T' for $\mathcal{T}(H)$. In order to see that (7.4) commutes, consider the following commutative diagram



The upper part commutes due to the left-hand triangle of (6.8), the lower triangle by Corollary 3.15, and the right-hand part is (7.9).

To see that (7.5) commutes, consider the following diagram



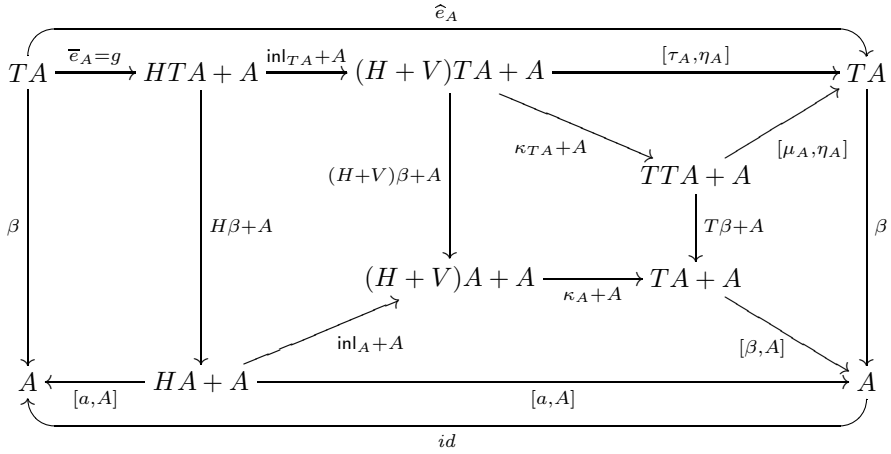
(7.10)

All of its inner parts commute. The upper part is Lemma 6.12, for the left-hand square, see (6.8), and for the right-hand part use that h is a monad morphism. That part (i) commutes follows from commutativity of the left-hand triangle of (6.8) and naturality. The remaining inner part commutes due to Corollary 3.15: $\mu \cdot \kappa T = \tau$. Thus, the outer shape of diagram (7.10) commutes. We obtain the equations

$$\begin{aligned} \beta \cdot \widehat{e}_A &= \widetilde{a} \cdot h_A \cdot \widehat{e}_A && \text{(see (7.9))} \\ &= \widetilde{a} \cdot h_A && \text{(see (7.10))} \\ &= \beta && \text{(see (7.9))} \end{aligned}$$

This completes the proof of part (i).

(ii) Let $(A, a, (\_)^*)$ be a cia. We show that the solution $e^\ddagger_A$ defined in (i) is unique. By Lemma 7.4, it suffices to prove that any evaluation morphism $\beta : \mathcal{T}(H+V)A \longrightarrow A$ for which diagrams (7.4) and (7.5) commute is a solution of g, see (7.7). To this end consider the following diagram, where we write $T = \mathcal{T}(H+V)$ for short once more:



Its outer shape commutes due to (7.5), the right-hand part since β is an Eilenberg-Moore algebra structure, the upper-right triangle follows from Corollary 3.15, and the lower right-hand part follows from (7.4). Thus, since all other parts are obviously commutative, the left-hand inner square commutes. But this shows that β is a solution of g, see (7.7). By the uniqueness of solutions, we have $\beta = g^*$. □

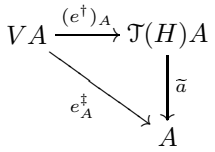
Definition 7.6. For any guarded RPS e and any Elgot algebra $(A, a, (\_)^*)$, let $e^\ddagger_A$ be the interpreted solution obtained from the proofs of Theorem 7.3 and Lemma 7.4 as stated below

$$e^\ddagger_A \equiv VA \xrightarrow{\mathsf{inr}} (H+V)A \xrightarrow{\kappa_A^{H+V}} \mathcal{T}(H+V)A \xrightarrow{g^*} A\,,$$

where g is the flat equation morphism of (7.7). We call this the *standard interpreted solution* of e in A.

Finally, we prove the "Fundamental Theorem of Algebraic Semantics", which establishes that uninterpreted and interpreted solutions are connected in the "proper way".

Theorem 7.7. *Let $(A, a, (\_)^*)$ be an Elgot algebra and consider its evaluation morphism $\widetilde{a} : \mathcal{T}(H)A \longrightarrow A$. Let e be any guarded recursive program scheme, let $e^\ddagger_A : VA \longrightarrow A$ be the standard interpreted solution of e in A of Theorem 7.3, and let $e^\dagger : V \longrightarrow \mathcal{T}(H)$ be the (uninterpreted) solution of Theorem 6.5. Then the triangle*



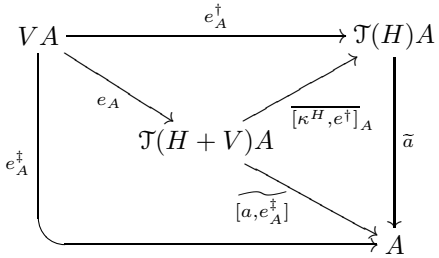
(7.11)

commutes.

Furthermore, the standard interpreted solution $e^\ddagger_A$ is uniquely determined by the commutativity of the above triangle.

Remark 7.8. Notice that $(e^\dagger)_A$ is the component at A of the natural transformation $e^\dagger$. And once again, $e^\ddagger_A$ is not the component at A of any natural transformation but merely a morphism from VA to A.

Proof. In fact, we have the commutative diagram



The lower left part commutes due to the definition of an interpreted solution, see (7.1); the top triangle is the definition of an uninterpreted solution, see (6.2); and the triangle on the right commutes by (7.9) since $h = \overline{[\kappa^H, e^\dagger]}$ and $\beta = \widetilde{[a, e_A^\ddagger]}$. The overall outside is what we want.

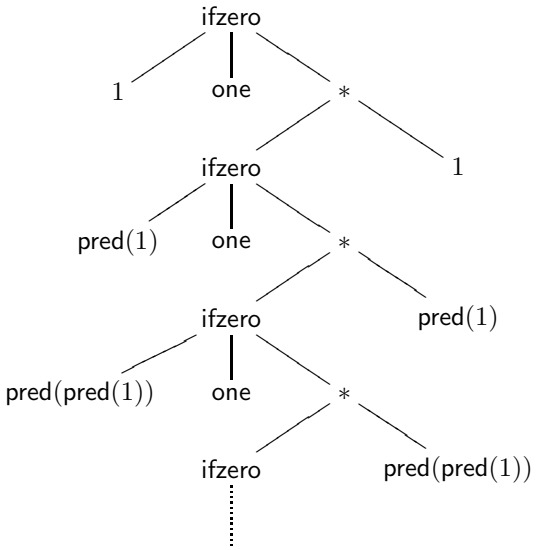
It is obvious that $e_A^\ddagger$ is uniquely determined by the commutativity of the triangle (7.11) as neither $\widetilde{a}$ nor $e^\dagger$ depend on $e_A^\ddagger$. □

7.1. **Interpreted Solutions in Computation Tree Elgot Algebras.** In this section, we work through some details concerning the factorial example of equation (1.2), repeated as

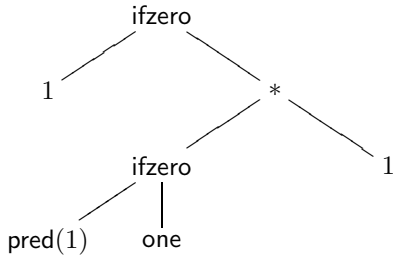
$$f(n) \approx \mathsf{ifzero}(n, \mathsf{one}, f(\mathsf{pred}(n) * n))$$

and studied further in Example 6.4(i). Recall that we work with the signature Σ containing a constant one, one unary symbol pred, a binary symbol $*$ and a ternary one ifzero. Let H_Σ be the associated endofunctor on Set. Let $\uparrow$ be an object which is not a natural number, and let $\mathbb{N}_\uparrow$ be the H_Σ-algebra with carrier $\{0, 1, 2, \ldots\} \cup \{\uparrow\}$ and whose operations are the strict extensions of the standard operations. (For example $\uparrow * 3 = \uparrow$ in this algebra.)

We shall use the computation tree Elgot algebra structure $(\mathbb{N}_\uparrow, a, (\_)^*)$ from Section 3.3. That structure used a particular element of the carrier set in connection with the conditional ifzero, and in our structure we take 0. Before looking at the interpreted solution to the factorial RPS, it might be useful to spell out the associated evaluation morphism $\widetilde{a} : \mathcal{T}(H_\Sigma)\mathbb{N}_\uparrow \longrightarrow \mathbb{N}_\uparrow$. Let t be a finite or infinite Σ-tree over $\mathbb{N}_\uparrow$; so the leaves of t might be labelled with natural number or $\uparrow$, but not with formal variables. Here is a pertinent example:



We got this by taking the uninterpreted solution of our RPS, as depicted in (6.4), and then substituting the number 1 for the formal variable n. Note that the nodes labelled ifzero have three children. Here is how we define $\widetilde{a}(t)$. We look for a finite subtree u of t with the property that if a node belongs to u and is labelled by a function symbol other than ifzero, then all its children belong to u as well; and if the node is labeled by ifzero, then either the first and second children belong to u, or else the first and third children do. For such a finite subtree u, we can evaluate the nodes in a bottom-up fashion using the H_Σ-algebra structure. We require that for a conditional node x, the first child evaluates to 0 (from our Elgot algebra structure) iff the second child is in u. If such a finite u exists, then we can read off an element of $\mathbb{N}_\uparrow$. This element is $\widetilde{a}(t)$. If no finite u exists, we set $\widetilde{a}(t) = \uparrow$. Returning to our example above, the finite subtree would be



And for our example tree t, $\widetilde{a}(t) = 1$.

We are now in a position to discuss the interpreted solution of our RPS. Recall that the signature Φ of recursively defined symbols contains only the unary symbol f. The corresponding signature functor is H_Φ, and $H_\Phi(\mathbb{N}_\uparrow)$ is the set $\{f(0), f(1), \ldots\} \cup \{f(\uparrow)\}$. The RPS itself is a natural transformation $e : H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$. The uninterpreted solution is the natural transformation $e^\dagger : H_\Phi \longrightarrow \mathcal{T}(H_\Sigma)$ corresponding to the tree shown in (6.4). We are concerned here with the interpreted solution $e^\ddagger_{\mathbb{N}_\uparrow} : H_\Phi(\mathbb{N}_\uparrow) \longrightarrow \mathbb{N}_\uparrow$ of our RPS. In light of the Fundamental Theorem 7.7, this is $\widetilde{a} \cdot (e^\dagger)_{\mathbb{N}_\uparrow}$. We show by an easy induction on $n \in N$ that this interpreted solution takes $f(n)$ to $n!$, and that it takes $f(\uparrow)$ to $\uparrow$.

We could also establish this same result directly, without Theorem 7.7. To do this, we follow the proof of Theorem 7.3. We turn our RPS e into a related natural transformation $\overline{e} : \mathcal{T}(H_\Sigma + H_\Phi) \longrightarrow H_\Sigma \mathcal{T}(H_\Sigma + H_\Phi) + Id$. Then $\overline{e}_{\mathbb{N}_\uparrow}$ is a flat equation morphism in the Elgot algebra $\mathbb{N}_\uparrow$, and its solution is the interpreted solution of our RPS. Here is a fragment of $\overline{e}_{\mathbb{N}_\uparrow}$:

$$\begin{aligned}
\underline{f(0)} &\approx \mathsf{ifzero}(\underline{0}, \underline{\mathsf{one}}, \underline{f(\mathsf{pred}(0)) * 0}) \\
\underline{f(1)} &\approx \mathsf{ifzero}(\underline{1}, \underline{\mathsf{one}}, \underline{f(\mathsf{pred}(1)) * 1}) \\
\underline{f(\mathsf{pred}(1)) * 1} &\approx \underline{\mathsf{ifzero}(\mathsf{pred}(1), \mathsf{one}, f(\mathsf{pred}(\mathsf{pred}(1))))} * \underline{1} \\
\underline{f(\mathsf{pred}(1))} &\approx \mathsf{ifzero}(\underline{\mathsf{pred}(1)}, \underline{\mathsf{one}}, \underline{f(\mathsf{pred}(\mathsf{pred}(1)) * \mathsf{pred}(1))}) \\
\underline{\mathsf{pred}(1)} &\approx \mathsf{pred}(\underline{1}) \\
\underline{\mathsf{one}} &\approx 1
\end{aligned}$$

One can see that for each natural number n, the solution to this flat equation morphism assigns to $\underline{f(n)}$ the number $n!$.

7.2. **Interpreted Solutions in CPO.** We shall show in this subsection that if we have $\mathcal{A} = \mathsf{CPO}$ as our base category, then interpreted solutions of guarded RPSs e in an Elgot algebra $(A, a, (\_)^*)$ are given as least fixed points of a continuous operator on a function space. In this way we recover denotational semantics from our categorical interpreted semantics of recursive program schemes.

Example 7.9. We study the RPS of equation (1.2) as formalized in Example 6.4(i). As we know, the intended interpreted solution is the factorial function on the natural numbers $\mathbb{N}$.

This time we turn the natural numbers into an object of CPO so as to obtain a suitable Elgot algebra in which we can find an interpreted solution of (1.2). Let $\mathbb{N}_\perp$ be the flat CPO obtained from the discretely ordered $\mathbb{N}$ by adding a bottom element $\perp$, i. e., $x \leq y$ iff $x = \perp$ or $x = y$. We equip $\mathbb{N}_\perp$ with the strict operations $\mathsf{one}_{\mathbb{N}_\perp}$, $\mathsf{pred}_{\mathbb{N}_\perp}$, and $*_{\mathbb{N}_\perp}$. These are all strict and hence continuous. In addition, we use the continuous function

$$\mathsf{ifzero}_{\mathbb{N}_\perp}(n, x, y) \quad = \quad \begin{cases} \perp & \text{if } n = \perp \\ x & \text{if } n = 0 \\ y & \text{else} \end{cases}$$

Indeed, this is what we saw in (3.4) for the computation tree semantics, except we write $\perp$ for $\uparrow$. Hence we have a continuous Σ-algebra with $\perp$. Therefore $\mathbb{N}_\perp$ is an Elgot algebra for $H_\Sigma : \mathsf{Set} \longrightarrow \mathsf{Set}$, see Example 3.8(iii).

The standard interpreted solution $e^\ddagger_{\mathbb{N}_\perp} : H_\Phi \mathbb{N}_\perp \longrightarrow \mathbb{N}_\perp$ will certainly be *some* function or other on $\mathbb{N}_\perp$. But how do we know that this function is the desired factorial function? Usually one would simply regard the RPS (1.2) itself as a continuous function R on $\mathsf{CPO}(\mathbb{N}_\perp, \mathbb{N}_\perp)$ acting as

$$f(\_) \longmapsto \mathsf{ifzero}_{\mathbb{N}_\perp}(\_, 1, f(\mathsf{pred}_{\mathbb{N}}(\_) *_{\mathbb{N}_\perp} \_), \_)\,;$$

i. e., R is the operator described in (1.5) in the introduction. That means that we interpret all the operation symbols of Σ in the algebra $\mathbb{N}_\perp$. The usual denotational semantics assigns to the formal equation (1.2) with the interpretation in $\mathbb{N}_\perp$ the least fixed point of R. Clearly this yields the desired factorial function. And it is not difficult to work out that the least fixed point of R coincided with the standard interpreted solution $e^\ddagger_{\mathbb{N}_\perp}$ obtained from Theorem 7.3. We shall do this shortly in greater generality.

In general, any recursive program scheme can be turned into a continuous operator R on the function space $\mathsf{CPO}(VA, A)$. Theorem 7.10 below shows that the least fixed point of R is the same as the interpreted solution obtained from Theorem 7.3.

We assume throughout this subsection that H, V and $H+V$ are locally continuous (and, as always, iteratable) endofunctors of CPO. We also consider a fixed guarded RPS $e : V \longrightarrow \mathcal{T}(H+V)$, and an H-algebra (A, a) with a least element $\bot$. By Example 3.8(ii), we know that this carries the structure of an Elgot algebra $(A, a, (\_)^*)$, where $(\_)^*$ assigns to every flat equation morphism a least solution. As before, we will use the notation $\widetilde{a} : \mathcal{T}(H)A \longrightarrow A$ for the induced evaluation morphism. Furthermore, for any continuous map $f : VA \longrightarrow A$ we have an Elgot algebra on A with structure $[a, f] : (H+V)A \longrightarrow A$. Due to Corollary 3.15, its evaluation morphism satisfies the equation

$$\widetilde{[a,f]} \cdot \kappa_A^{H+V} = [a, f]\,. \tag{7.12}$$

Theorem 7.10. *The following function R on $\mathsf{CPO}(VA, A)$*

$$f \longmapsto VA \xrightarrow{e_A} \mathcal{T}(H+V)A \xrightarrow{\widetilde{[a,f]}} A \tag{7.13}$$

is continuous. Its least fixed point is the standard interpreted solution $e_A^\ddagger : VA \longrightarrow A$ of Theorem 7.3.

Proof. (i) To see the continuity of R is suffices to prove that the function

$$\widetilde{(\_)} : \mathsf{CPO}(HA, A) \longrightarrow \mathsf{CPO}(\mathcal{T}(H)A, A)$$

is continuous. Let us write T for $\mathcal{T}(H)$. Recall that for any continuous map $a : HA \longrightarrow A$, the evaluation morphism $\widetilde{a}$ is the least solution of the flat equation morphism $\alpha_A : TA \longrightarrow HTA + A$, i. e., $\widetilde{a}$ is the least fixed point of the continuous function

$$F(a, -) : \mathsf{CPO}(TA, A) \longrightarrow \mathsf{CPO}(TA, A) \qquad f \longmapsto [a, A] \cdot (Hf + A) \cdot \alpha_A\,.$$

Observe that F is continuous in the first argument a, and so F is a continuous function on the product $\mathsf{CPO}(HA, A) \times \mathsf{CPO}(TA, A)$. It follows from standard arguments that taking the least fixed point in the second argument yields a continuous map $\mathsf{CPO}(HA, A) \longrightarrow \mathsf{CPO}(TA, A)$. But this is precisely the desired one $\widetilde{(\_)}$.

(ii) We prove that $e_A^\ddagger$ is the least fixed point of R. Notice that the least fixed point of R is the join t of the following increasing chain in $\mathsf{CPO}(VA, A)$:

$$t_0 = \mathsf{const}_\bot : VA \longrightarrow A, \qquad t_{i+1} \equiv VA \xrightarrow{e_A} \mathcal{T}(H+V)A \xrightarrow{\widetilde{[a,t_i]}} A, \qquad \text{for } i \geq 0.$$

Furthermore, recall that the interpreted solution $e_A^\ddagger$ is defined by (7.6) as

$$e_A^\ddagger \equiv \beta \cdot \kappa_A^{H+V} \cdot \mathsf{inr}\,,$$

where $\beta = g^*$ is the least solution of the flat equation morphism g which is obtained from the component at A of the $\mathcal{H}$-coalgebra $\overline{e}$, see Lemma 6.9 and Theorem 7.3. By Example 3.8(ii), the solution β of g is the join of the chain

$$\beta_0 = \mathsf{const}_\bot, \qquad \beta_{i+1} = [a, A] \cdot H(\beta_i + A) \cdot g, \quad \text{for } i \geq 0.$$

Observe that $e_A^\ddagger$ is a fixed point of R, see (7.1). Thus, we have $t \sqsubseteq e_A^\ddagger$. To show the reverse inequality we will prove by induction on i the inequalities

$$\beta_i \sqsubseteq \widetilde{[a,t]}\,, \quad i \in \mathbb{N}\,. \tag{7.14}$$

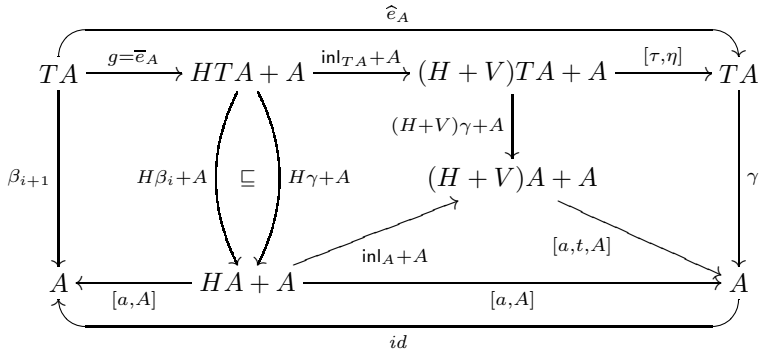
This implies that $\beta \sqsubseteq \widetilde{[a,t]}$ and therefore

$$e_A^\ddagger = \beta \cdot \kappa_A^{H+V} \cdot \mathsf{inr} \sqsubseteq \widetilde{[a,t]} \cdot \kappa_A^{H+V} \cdot \mathsf{inr} = t\,,$$

where the last equality follows from (7.12).

We complete the proof with the induction proof to establish (7.14). The base case is clear. For the induction step we write T for $\mathcal{T}(H+V)$ and γ as a short notation for $\widetilde{[a,t]}$ and we consider the following

diagram



Its upper part commutes due to Lemma 6.12, the left-hand part is the definition of β_{i+1}, and the inequality follows from the induction hypothesis. For the right-hand part recall that $\gamma = \widetilde{[a,t]} : TA \longrightarrow A$ is a homomorphism of Elgot algebras, see Theorem 3.13. Hence, γ is an algebra homomorphism by Proposition 3.6. Finally, the remaining parts of the above diagram clearly commute. Thus we obtain the inequality $\beta_{i+1} \sqsubseteq \gamma \cdot \widehat{e}_A$. Finally, since t is a fixed point of R it is an interpreted solution of e in A. By Lemma 7.4, it follows that $\gamma \cdot \widehat{e}_A = \gamma$, and this establishes the desired inequality. □

Remark 7.11. The result of Theorem 7.10 implies a concrete formula

$$e^\ddagger_A = \bigsqcup_{n<\omega} e^\ddagger_n$$

for the interpreted solution of the guarded RPS e in the continuous algebra A. In fact, the least fixed point of R is the join of the ascending chain

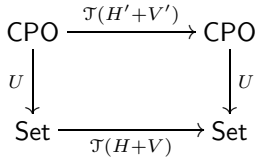
$$\bot \sqsubseteq R(\bot) \sqsubseteq R^2(\bot) \sqsubseteq \cdots$$

where $\bot = \mathsf{const}_\bot$ is the least element of $\mathsf{CPO}(VA, A)$. Thus, with $e^\ddagger_0 = \mathsf{const}_\bot$ and

$$e^\ddagger_{n+1} \equiv VA \xrightarrow{e_A} \mathcal{T}(H+V)A \xrightarrow{\widetilde{[a,e^\ddagger_n]}} A$$

we obtain the above formula for $e^\ddagger_A$.

Remark 7.12. Suppose that H, V and $H+V$ are iteratable endofunctors of Set, which have locally continuous liftings H' and V' to CPO. Then we have a commutative square



see Example 2.5(iv). Furthermore, assume that the guarded RPS $e : V \longrightarrow \mathcal{T}(H+V)$ has a lifting $e' : V' \longrightarrow \mathcal{T}(H'+V')$; i. e., a natural transformation e' such that $U * e' = e * U$. Now consider any CPO-enrichable H-algebra (A, a) as an Elgot algebra, see Example 3.8(iii). Then we can apply Theorem 7.10 to obtain the standard interpreted solution $e^\ddagger_A$ of e in the algebra A as a least fixed point of the above function R of (7.13).

Example 7.13.

(i) Suppose we have signatures Σ and Φ. Then the signature functors H_Σ and H_Φ have locally continuous liftings H'_Σ and H'_Φ. Since the lifting of $H_\Sigma + H_\Phi$ is a lifting of $H_{\Sigma+\Phi}$ we know that $\mathcal{T}(H'_\Sigma + H'_\Phi)$ assigns to any cpo X the algebra $T_{\Sigma+\Phi}X$ with the cpo structure induced by X, see Example 2.5(v). More precisely, to compare a tree t to a tree s replace all leaves labelled by a variable from X by a leaf labelled by some extra symbol $\star$ to obtain relabelled trees t' and s'. Then t is less than s iff t' and s' are isomorphic as labelled trees, and for any leaf of t labelled by a variable x the corresponding leaf in s is labelled by a variable y with $x \sqsubseteq y$ in X.

Now consider any system as in (2.3) which is in Greibach normal form, and form the associated guarded RPS $e : H_\Phi \longrightarrow T_{\Sigma+\Phi}$. Then e has a lifting $e' : H'_\Phi \longrightarrow \mathcal{T}(H'_\Sigma + H'_\Phi)$. In fact, for any cpo X the component $e'_X = e_X : H_\Phi X \longrightarrow T_{\Sigma+\Phi}X$ is a continuous map since the order in $H_\Phi X$ is given similarly as for $T_{\Sigma+\Phi}X$ on the level of variables only: $(f, \vec{x}) \sqsubseteq (g, \vec{y})$ holds for elements of $H_\Phi X$ if $f = g \in \Phi_n$ and $x_i \sqsubseteq y_i$, $i = 1, \ldots, n$, holds in X.

Let (A, a) be a CPO-enrichable H_Σ-algebra; i. e., a continuous Σ-algebra with a least element $\bot$. We wish to consider the continuous function R on $\mathsf{CPO}(H_\Phi A, A)$ which assigns to any continuous algebra structure $\varphi : H_\Phi A \longrightarrow A$ the algebra structure $R(\varphi) = \widetilde{[a, \varphi]} \cdot e'_A$. The structure $R(\varphi) : H_\Phi A \longrightarrow A$ gives to each n-ary operation symbol f of Φ the operation $t^f_A : A^n \longrightarrow A$ which is obtained as follows: take the term t^f provided by the right-hand side of f in our given RPS, then interpret all operation symbols of Σ in t^f according to the given algebraic structure a and all operation symbols of Φ according to φ; the action of t^f_A is evaluation of that interpreted term.

Theorem 7.10 states that the standard interpreted solution $e^\ddagger_A$ of e in the algebra A can be obtained by taking the least fixed point of R; in other words the standard interpreted solution $e^\ddagger_A$ gives the usual denotational semantics.

(ii) Apply the previous example to the RPS of Example 6.4(i). Then Theorem 7.10 states that the interpreted solution of the RPS (1.2) in the Elgot algebra $\mathbb{N}_\bot$ is obtained as the least fixed point of the function R of Example 7.9. That is, the standard interpreted solution gives the desired factorial function.

(iii) Recall the guarded RPS $e : V \longrightarrow \mathcal{T}(H + V)$ from Example 6.4(ii) whose uninterpreted solution we have described in Example 6.6(ii). Consider again the algebra $\mathbb{N}_\bot$ together with the following two operations:

$$F_{\mathbb{N}_\bot}(x, y, z) = \begin{cases} x & \text{if } x = y \\ z & \text{else} \end{cases} \qquad G_{\mathbb{N}_\bot}(x) = \begin{cases} \lfloor \frac{x}{2} \rfloor & \text{if } x \in \mathbb{N} \\ \bot & x = \bot \end{cases} \tag{7.15}$$

Since the first operation obviously satisfies $F_{\mathbb{N}_\bot}(x, y, z) = F_{\mathbb{N}_\bot}(y, x, z)$ we have defined an H-algebra. It is not difficult to check that the set functor H has a locally continuous lifting H' to CPO and that $\mathbb{N}_\bot$ is a continuous H'-algebra. In fact, the existence of the lifting H' follows from the fact that the unordered pair functor $V : \mathsf{Set} \longrightarrow \mathsf{Set}$ can be lifted to CPO; the lifting assigns to a cpo $(X, \leq)$ the set of unordered pairs with the following order: $\{ x, y \} \sqsubseteq \{ x', y' \}$ iff either $x \leq x'$ and $y \leq y'$, or $x \leq y'$ and $y \leq x$. Thus, we have defined an Elgot algebra for $H : \mathsf{Set} \longrightarrow \mathsf{Set}$, see Example 3.8(iii). The standard interpreted solution $e^\ddagger_{\mathbb{N}_\bot} : V\mathbb{N}_\bot \longrightarrow \mathbb{N}_\bot$ is given by one commutative binary operation $\varphi_{\mathbb{N}_\bot}$ on $\mathbb{N}_\bot$. We leave it to the reader to verify that for natural numbers n and m, $\varphi_{\mathbb{N}_\bot}(n, m)$ is the natural number represented by the greatest common prefix in the binary representation of n and m, e. g., $\varphi_{\mathbb{N}_\bot}(12, 13) = 6$. Notice that we do not have to prove separately that $\varphi_{\mathbb{N}_\bot}$ is commutative. In Example 6.4(ii) we have encoded that extra property directly into the RPS e so that any solution must be commutative.

(iv) Least fixed points are RPS solutions. Let A be a poset with joins of all subsets which are at most countable, and let $f : A \longrightarrow A$ be a function preserving joins of ascending chains. Take f and binary joins to obtain an algebra structure on A of the signature functor $H_\Sigma X = X + X \times X$ expressing a binary operation symbol F and a unary one G. Obviously, this functor has a lifting $H' : \mathsf{CPO} \longrightarrow \mathsf{CPO}$ and A is a CPO-enrichable algebra, i. e., A is an Elgot algebra. Turn the formal equations (1.1) into a recursive program scheme $e : H_\Phi \longrightarrow \mathcal{T}(H_\Sigma + H_\Phi)$ as demonstrated in Section 2.3. The RPS e has a lifting $e' : V' \longrightarrow \mathcal{T}(H' + V')$, where V' denotes the lifting of H_Φ. The standard interpreted solution $e^\ddagger_A : V'A \longrightarrow A$ gives two continuous functions φ_A and ψ_A on A. Clearly, we have $\varphi_A(a) = \bigvee_{n \in \mathbb{N}} f^n(a)$, and in particular $\varphi_A(\bot)$ is the least fixed point of f.

7.3. **Interpreted Solutions in CMS.** Recall the category CMS of complete metric spaces from Example 2.2(iii), and let $H, V : \mathsf{CMS} \longrightarrow \mathsf{CMS}$ be contracting endofunctors. We shall show in this subsection that for any guarded RPS $e : V \longrightarrow \mathcal{T}(H + V)$ we can find a unique interpreted solution in any non-empty H-algebra A. More precisely, assume that we have such a guarded RPS e, and let (A, a) be a non-empty H-algebra. Then A is a cia, and, in particular it carries the structure of an Elgot algebra. Notice that for any non-expanding map $f : VA \longrightarrow A$ we obtain an algebra structure $[a, f] : (H + V)A \longrightarrow A$, thus we have the evaluation morphism

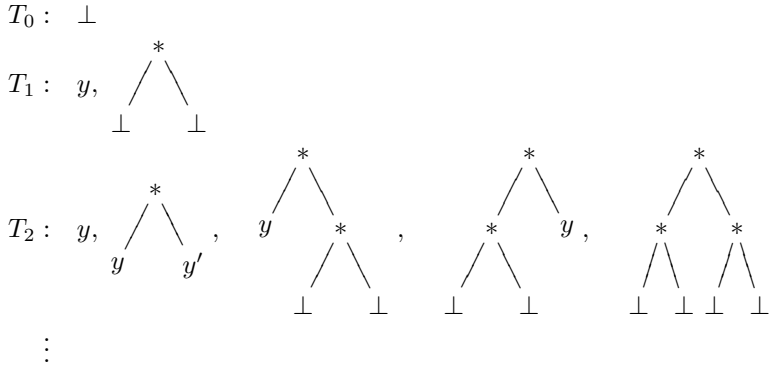
$$\widetilde{[a, f]} : \mathcal{T}(H + V)A \longrightarrow A\,.$$

As in CPO, the RPS e induces a function R on $\mathsf{CMS}(VA, A)$, see (7.13). The standard procedure for obtaining an interpreted solution would be to prove that R is a contracting map, and then invoke Banach's Fixed Point theorem to obtain a unique fixed point of R. Here we simply apply Theorem 7.3. Notice, however, that we cannot completely avoid Banach's Fixed Point theorem: it is used in the proof that final coalgebras exist for contracting functors, see [ARe].

Corollary 7.14. *The unique interpreted solution $e^\dagger_A : VA \longrightarrow A$ of e in A as obtained in Theorem 7.3 is the unique fixed point of the function R on $\mathsf{CMS}(VA, A)$ defined by (7.13).*

Proof. In fact, being a fixed point of R is equivalent to being an interpreted solution of e in the cia A, whose unique existence we have by Theorem 7.3. □

Remark 7.15. Let H_Σ be a signature functor on Set and denote by H' a lifting to CMS as described in Example 3.2(v). For a complete metric space Y the final coalgebra $\mathcal{T}(H')Y$ of $H'(\_) + Y$ is the set $T_\Sigma Y$ of all Σ-trees over Y equipped with a suitable complete metric. This metric can be described as follows. Recall from [ARe] that $\mathcal{T}(H')Y$ is obtained as T_ω after ω steps of the final coalgebra chain for $H'(\_) + Y$, see Construction 2.3. That means the metric on $T_\Sigma Y$ is the smallest metric such that all projections $t_{\omega,i} : T_\Sigma Y = T_\omega \longrightarrow T_i$ are non-expanding. We illustrate this with an example adapted from [ARe]. Let $H_\Sigma X = X \times X$ be the functor expressing one binary operation symbol $*$. Then we can represent $T_0 = 1$ by a single node tree labelled with $\bot$ and $T_{i+1} = T_i \times T_i + Y$ by trees which are either single node trees labelled in Y, or which are composed by joining two trees from T_i with a root labelled by $*$:



The distance on T_1 is that of Y for single node trees and 1 otherwise. The distance on T_2 is again that of Y between single node trees, and 1 between single node trees and all other trees. Furthermore, the distance between trees of different shapes is $\frac{1}{2}$, and finally, $d_{T_2}(y * y', z * z') = \frac{1}{2} \max\{\, d_Y(y, z), d_Y(y', z') \,\}$ as well as $d_{T_2}(y * t, y' * t) = d_{T_2}(t * y, t * y') = \frac{1}{2} d_Y(y, y')$, where $t = \bot * \bot$, etc. In general, the distance on T_{i+1} is that of Y between single node trees, it is 1 between single node trees and trees of height at least 1, and otherwise we have $d_{T_{i+1}}(s * t, s' * t') = \frac{1}{2} \max\{\, d_{T_i}(s, s'), d_{T_i}(t, t') \,\}$. For the metric on $T_\Sigma Y$, we have

$$d_{T_\Sigma Y}(s_1, s_2) = \sup_{i<\omega} d_{T_i}(t_{\omega,i}(s_1), t_{\omega,i}(s_2)).$$

This is the smallest metric for which the projections are non-expanding. (One may also verify directly that this definition gives a complete metric space structure and that $H'(\_) + Y$ preserves the limit, so that we indeed have a final coalgebra.) Finally notice that the metric of $T_\Sigma Y$ depends on the choice of the lifting H'. For example, if we lift the functor H_Σ as $H'(X, d) = (X^2, \frac{1}{3} d_{\max})$, the factor $\frac{1}{2}$ would have to be replaced by $\frac{1}{3}$ systematically.

Example 7.16.

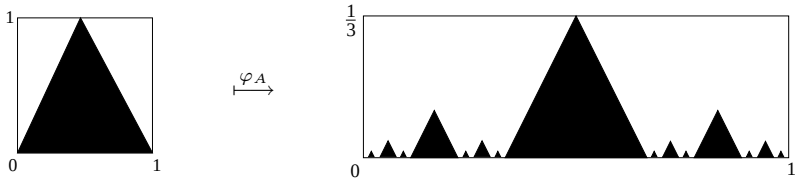
(i) Consider the endofunctor $H' : \mathsf{CMS} \longrightarrow \mathsf{CMS}$ obtained by lifting the signature functor $H_\Sigma X = X \times X + X$ expressing a binary operation F and a unary one G as described in Example 2.5(v). The Euclidean interval $I = [0, 1]$ together with the operations $F(x, y) = \frac{x+y}{4}$ and $G(x) = \frac{\sin(x)}{2}$ is an H'-algebra, whence a cia. Use only the first equation in (1.1) to obtain a guarded RPS $e : Id \longrightarrow \mathcal{T}(H_\Sigma + Id)$ where Id expresses the unary operation symbol φ. Let V' be contracting lifting of Id with a contraction factor of $\varepsilon = \frac{1}{2}$. Then e gives rise to a guarded RPS $e' : V' \longrightarrow \mathcal{T}(H' + V')$ in CMS. The unique interpreted solution of e' in I consists of a function $\varphi_I : I \longrightarrow I$ satisfying $\varphi_I(x) = \frac{1}{4}(x + \varphi_I(\frac{1}{2} \sin x))$, that is, φ_I is the unique function f satisfying (1.3).

(ii) Self-similar sets are solutions of interpreted program schemes. Recall from Example 3.2(v) that for any complete metric space (X, d) we obtain the complete metric space $(C(X), h)$ of all non-empty compact subspaces of X with the Hausdorff metric. Furthermore, contractive mappings of X yield structures of cias on $C(X)$. Now consider the functor H' on CMS with $H(X, d) = (X^3, \frac{1}{3} d_{\max})$, where $d_{\max}$ is the maximum metric. It is a lifting of the signature functor H_Σ on Set expressing one ternary operation α. Let $A = [0, 1] \times [0, 1]$, be equipped with the usual Euclidean metric. Consider the contracting maps $f(x, y) = (\frac{1}{3}x, \frac{1}{3}y)$, $g(x, y) = (\frac{1}{3}x + \frac{1}{3}, \frac{1}{3}y)$, and $h(x, y) = (\frac{1}{3}x + \frac{2}{3}, \frac{1}{3}y)$ of A.

Then it follows that $\alpha_A : C(A)^3 \longrightarrow C(A)$ with $\alpha(D, E, F) = f[D] \cup g[E] \cup h[F]$ is a $\frac{1}{3}$-contracting map, whence a structure of an H'-algebra. The formal equation

$$\varphi(x) \approx \alpha(\varphi(x), x, \varphi(x))$$

gives rise to a guarded RPS $e : Id \longrightarrow \mathcal{T}(H_\Sigma + Id)$, where the identity functor expresses the operation φ. If we take the lifting of Id to CMS which is given by $V'(X, d) = (X, \frac{1}{3}d)$, then e gives rise to a natural transformation $e' : V' \longrightarrow \mathcal{T}(H' + V')$. Its interpreted solution in the algebra $C(A)$ is a $\frac{1}{3}$-contracting map $\varphi_A : C(A) \longrightarrow C(A)$ which maps a non-empty compact subspace U of A to a space of the following form: $\varphi_A(U)$ has three parts, the middle one is a copy of U scaled by $\frac{1}{3}$, and the left-hand and right-hand one look like copies of the whole space $\varphi_A(U)$ scaled by $\frac{1}{3}$. For example, we have the assignment



(iii) Coming back to Example 3.2(vi) let us consider $(C(I), \alpha_I)$, where $I = [0, 1]$ is the Euclidean interval, $C(I)$ is the set of all non-empty closed subsets of I, and α_I is the structure of a cia arising from $f(x) = \frac{1}{3}x$ and $g(x) = \frac{1}{3}x + \frac{2}{3}$ as described is Example 3.2(v). The formal equation

$$\varphi(x) \approx \alpha(\varphi(x), x)$$

gives similarly as in (i) above a guarded RPS $e : Id \longrightarrow \mathcal{T}(H_\Sigma + Id)$, where $H_\Sigma X = X \times X$ now expresses the binary operation α. Again, we have liftings $V'(X, d) = (X, \frac{1}{3}d)$ and $H'(X, d) = (X^2, \frac{1}{3}d_{\max})$ of Id and H_Σ, respectively. So the RPS e lifts to the guarded RPS $e' : V' \longrightarrow \mathcal{T}(H' + V')$ in CMS. Its unique interpreted solution is given by the $\frac{1}{3}$-contracting map $\varphi_I : C(I) \longrightarrow C(I)$ satisfying $\varphi_I(t) = \alpha_I(\varphi_I(t), t) = f[\varphi_I(t)] \cup g[t]$ for every non-empty closed subset t of the interval I.

8. Conclusions and Future Work

We have presented a general and conceptually clear way of treating the uninterpreted and the interpreted semantics of recursive program schemes in a category theoretic setting. For this we have used recent results on complete Elgot algebras and results from the theory of coalgebras. We have shown that our theory readily specializes to the classical setting yielding denotational semantics using complete partial orders or complete metric spaces. We have presented new applications of recursive program scheme solutions including fractal self-similarity and also applications which cannot be handled by the classical methods; defining operations satisfying equations like commutativity. Another new application, recursively defined functions on non-wellfounded sets, will be treated in a future paper.

Now one must go forward in reinventing algebraic semantics with category theoretic methods. We strongly suspect that there is much to be said about the relation of our work to operational semantics. We have not investigated higher-order recursive program schemes using our tools, and it would be good to know whether our approach applies in that area as well. The paper [MU] addresses variable binding and infinite terms coalgebraically, and this may well be relevant. Back to the classical theory, one of the main goals of the original theory is to serve as a foundation for program equivalence. It is not difficult to prove the soundness of fold/unfold transformations in an algebraic way using our semantics; this was done in [Mo_2] for uninterpreted schemes. We study the equational properties of our very general formulation of recursion in [MM_2]. One would like more results of this type. The equivalence of interpreted schemes in the natural numbers is undecidable, and so one naturally wants to study the equivalence of interpreted schemes in *classes of interpretations*. The classical theory proposes classes of interpretations, many of which are defined on ordered algebras, see [G]. It would be good to revisit this part of the classical theory to see whether Elgot algebras suggest tractable classes of interpretations.

Another path of future research is the study of algebraic trees with categorical methods. In the setting of trees over a signature Σ the solutions of recursive program schemes form the theory of algebraic trees, a subtheory of the theory of all trees on Σ. Moreover, algebraic trees are closed under second-order substitution and they form an iterative theory in the sense of Elgot [E]. Similar results should be possible to obtain in our generalized categorical setting. First promising steps in the direction of categorically studying algebraic trees have been taken in [AMV_4].

Acknowledgments

We are grateful to Jiří Adámek for many conversations on this and related matters, and for his enthusiasm for our project.

References

[A] Peter Aczel, *Non-Well-Founded Sets*. CSLI Lecture Notes Number 14, CSLI Publications, Stanford, 1988.

[AAV] P. Aczel, J. Adámek and J. Velebil, A Coalgebraic View of Infinite Trees and Iteration, *Electron. Notes Theor. Comput. Sci.* 44 (2001), no. 1, 26 pp.

[AAMV] P. Aczel, J. Adámek, S. Milius and J. Velebil, Infinite Trees and Completely Iterative Theories: A Coalgebraic View, *Theoret. Comput. Sci.* 300 (2003), 1–45.

[AK] J. Adámek and V. Koubek, On the greatest fixed point of a set functor, *Theoret. Comput. Sci.* 150 (1995), 57–75.

[AM] J. Adámek and S. Milius, Terminal Coalgebras and Free Iterative Theories, accepted for publication in *Inform. and Comput.*

[AMV$_1$] J. Adámek, S. Milius and J. Velebil, Free Iterative Theories: A Coalgebraic View, *Math. Structures Comput. Sci.* 13 (2003), 259–320.

[AMV$_2$] J. Adámek, S. Milius and J. Velebil, From Iterative Algebras to Iterative Theories, extended abstract appeared in *Electron. Notes Theor. Comput. Sci.* 106 (2004), 3–24, full version submitted and available at the URL `http://www.iti.cs.tu-bs.de/~milius`.

[AMV$_3$] J. Adámek, S. Milius and J. Velebil, On coalgebra based on classes, *Theoret. Comput. Sci.* 316 (2004), 3-23.

[AMV$_3$] J. Adámek, S. Milius and J. Velebil, Elgot Algebras, preprint, 2005, available at the URL `http://www.iti.cs.tu-bs.de/~milius`, extended abstract to appear in *Electron. Notes Theor. Comput. Sci.*

[AMV$_4$] J. Adámek, S. Milius and J. Velebil, Algebraic Trees? Who knows. . . , working draft, May 2005.

[AP] J. Adámek and H. E. Porst, On tree coalgebras and coalgebra presentations, *Theoret. Comput. Sci.* 311 (2004), 257–283.

[ARe] J. Adámek and J. Reitermann, Banach's Fixed-Point Theorem as a Base for Data-Type Equations, *Appl. Categ. Structures* 2 (1994), 77–90.

[AT] J. Adámek and V. Trnková, *Automata and Algebras in Categories*. Kluwer Academic Publishers, 1990.

[ARu] P. America and J. J. M. M. Rutten, Solving Reflexive Domain Equations in a Category of Complete Metric Spaces, *J. Comput. System Sci.* 39 (1989), 343–375.

[AN] A. Arnold and M. Nivat, The metric space of infinite trees. Algebraic and topological properties, *Fund. Inform.* III, no. 4 (1980), 445–476.

[B] M. F. Barnsley, *Fractals Everywhere*, Academic Press 1988.

[Ba] M. Barr, Terminal coalgebras in well-founded set theory, *Theoret. Comput. Sci.* 114 (1993), 299–315.

[BM] J. Barwise and L. S. Moss, *Vicious Circles*, CSLI Publications, Stanford, 1996.

[Bl] S. L. Bloom, All Solutions of a System of Recursion Equations in Infinite Trees and Other Contraction Theories, *J. Comput. System Sci.* 27 (1983), 225–255.

[BE] S. L. Bloom and Z. Ésik, *Iteration Theories: The equational logic of iterative processes,* EATCS Monographs on Theoretical Computer Science, Berlin: Springer-Verlag (1993).

[Bo] F. Borceux, *Handbook of Categorical Algebra 2: Categories and Structures*, Vol. 51 of Encyclopedia of Mathematics and its Applications, Cambridge University Press, 1994.

[C] B. Courcelle, Fundamental properties of infinite trees, *Theoret. Comput. Sci.* 25 (1983), no. 2, 95–169.

[E] C. C. Elgot, Monadic Computation and Iterative Algebraic Theories, in: *Logic Colloquium '73* (eds: H. E. Rose and J. C. Shepherdson), North-Holland Publishers, Amsterdam, 1975.

[EBT] C. C. Elgot, S. L. Bloom and R. Tindell, On the Algebraic Structure of Rooted Trees, *J. Comput. System Sci.* 16 (1978), 361–399.

[GLMP$_1$] N. Ghani, C. Lüth, F. De Marchi and A. J. Power, Algebras, coalgebras, monads and comonads, *Electron. Notes Theor. Comput. Sci.* 44 (2001), no. 1, 18 pp.

[GLMP$_2$] N. Ghani, C. Lüth, F. De Marchi and A. J. Power, Dualising initial algebras, *Math. Structures Comput. Sci.* 13 (2003), no. 2, 349–370.

[GLM] N. Ghani, C. Lüth and F. De Marchi, Solving Algebraic Equations using Coalgebra, *Theor. Inform. Appl.* 37 (2003), 301–314.

[ADJ] J. A. Goguen, S. W. Thatcher, E. G. Wagner and J. B. Wright, Initial Algebra Semantics and Continuous Algebras, *J. ACM* 24 (1977), 68–95.

[G] I. Guessarian, *Algebraic Semantics*. Lecture Notes in Comput. Sci., Vol. 99, Springer, 1981.

[L] J. Lambek, A Fixpoint Theorem for Complete Categories, *Math. Z.* 103 (1968), 151–161.

[L$_1$] T. Leinster, General self-similarity: an overview, e-print math.DS/0411343 v1.

[L$_2$] T. Leinster, A general theory of self-similarity I, e-print math.DS/041344.

[L$_3$] T. Leinster, A general theory of self-similarity II, e-print math.DS/0411345.

[ML] S. MacLane, *Categories for the working mathematician*, 2nd edition, Springer Verlag, 1998.

[MU] R. Matthes and T. Uustalu, Substitution in Non-Wellfounded Syntax with Variable Binding. In H. P. Gumm, (ed.), *Electron. Notes Theor. Comput. Sci.*, 82 (2003), no. 1, 15 pp.

[Mi$_1$] S. Milius, On Iteratable Endofunctors, *Electron. Notes Theor. Comput. Sci.* 69 (2002), 18 pp.

[Mi$_2$] S. Milius, Completely Iterative Algebras and Completely Iterative Monads, *Inform. and Comput.* 196 (2005), 1–41.

[MM] S. Milius and L. S. Moss, The Category Theoretic Solution of Recursive Program Schemes, extended abstract, in Proceedings of the 1st International Conference on Algebra and Coalgebra in Computer Science (CALCO 2005), *Lecture Notes in Comput. Sci.* 3629 (2005), 293–312.

[MM_2] S. Milius and L. S. Moss, Equational Properties of Solutions to Recursive Program Schemes, working draft.
[Mo_1] L. S. Moss, Parametric Corecursion, *Theoret. Comput. Sci.* 260 (2001), no. 1–2, 139–163.
[Mo_2] L. S. Moss, The Coalgebraic Treatment of Second-Order Substitution and Uninterpreted Recursive Program Schemes, preprint, 2002.
[Mo_3] L. S. Moss, Uniform Functors on Sets, submitted.
[N] M. Nivat, On the Interpretation of Recursive Polyadic Program Schemes, *Symposia Mathematica XV* (1975), 255–281.
[W] J. Worrell, On the Final Sequence of a Finitary Set Functor, *Theoret. Comput. Sci.* 338 (2005), 184–199.

Institute of Theoretical Computer Science, Technical University, Braunschweig, Germany
E-mail address: milius@iti.cs.tu-bs.de

Department of Mathematics, Indiana University, Bloomington, IN, USA
E-mail address: lsm@cs.indiana.edu