The Category Theoretic Solution of Recursive Program Schemes*

Stefan Milius[1] and Lawrence S. Moss[2]

[1] Institute of Theoretical Computer Science, Technical University, Braunschweig, Germany milius@iti.cs.tu-bs.de
[2] Department of Mathematics, Indiana University, Bloomington, IN, USA, lsm@cs.indiana.edu

Abstract. This paper provides a general account of the notion of recursive program schemes, their uninterpreted and interpreted solutions, and related concepts. 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 recursion, e. g., 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 at all. For example, the classical Cantor two-thirds set falls out as an interpreted solution (in our sense) of a recursive program scheme. In this short version of our paper we can only sketch some proofs.

1 Introduction

The theory of *recursive program schemes* is a topic at the heart of semantics. One takes a system of equations such as

$$\varphi(x) \approx F(x, \varphi(Gx)) \qquad \psi(x) \approx F(\varphi(Gx), GGx)) \tag{1.1}$$

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, where one also has an algebra A with operations for all the given operation symbols, and *uninterpreted* schemes.

This paper presents a generalization of the classical theory based on *Elgot algebras* and *coalgebras*. The point in a nutshell is that knowing that the infinite

* The full version of the current paper containing all proofs can be found at the URL http://www.iti.cs.tu-bs.de/~milius

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 [1, 28]. One does not need to assume any metric or order to study infinite trees: the finality principle is sufficient. However, to extend the result to second-order substitution, we need additional ideas. 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 [5]. 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. 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. Finally, we also provide examples of recursive program schemes and their solutions which cannot be treated with the classical theory: recursive definitions of operations satisfying equations like commutativity, examples in non-well founded sets (solving $x = \{\mathcal{P}_{\mathrm{f}}(x)\}$), and fractal self-similarity (the Cantor set c solves $c = \frac{1}{3}c \cup (\frac{2}{3} + \frac{1}{3}c)$). In some cases, one could use the classical theory (but even in those our application appears new). Our overall claim is that we have a unified view of solution principles for a large class of implicit definitions including recursive program schemes.

Our theory is based on notions from category theory (monads, Eilenberg-Moore algebras, Elgot algebras) and coalgebra (finality, solution theorems, completely iterative algebras). Our 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 $H : \mathcal{A} \to \mathcal{A}$ such that for all objects X a final coalgebra TX of $H(\_) + X$ exists. 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.

Related Work. The classical theory of recursive program schemes is compactly presented by Guessarian [20]. There one finds results on uninterpreted solutions of program schemes and interpreted ones in continuous algebras.

The realization that solutions of formal recursive equations can be studied with coalgebraic methods goes back to the second author [28], and appears also in the work of Aczel et al. [1], which generalizes and extends results of Elgot et al. [17, 18], see also [26] of the first author. From [1, 26] it also follows how to generalize *second order substitution* of infinite trees (see Courcelle [16]) to final coalgebras. The types of recursive equations studied in [28, 1] 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. [19] 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 [1] is new here.

Complete metric spaces as a basis for the semantics of recursive program schemes have been studied by Arnold and Nivat, see e. g. [10]. Bloom [14] 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 in [9, 7]. We build on this with our treatment of self-similar objects. These have also recently been studied in a categorical framework by Leinster, see [22–24]. The example in this paper uses standard results on complete metric spaces, see e.g. [11].

In this short abstract we can only present sketches of proofs and we are forced to omit most of the technical detail. We refer the interested reader to the full version [27] of this paper.

2 Preliminaries

Assumption 2.1. 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.

These are fairly mild conditions. 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. In the concrete categories we consider here there are stronger yet much nicer conditions that ensure iteratability:

Examples 2.2. (i) In the category Set of sets and maps any accessible (equivalently, bounded, see [6]) endofunctor is iteratable, see [12].
(ii) Consider the category CPO of complete partial orders; i. e., posets (not necessarily with a least element) in which every ascending chain has a join, and continuous maps. Notice that coproducts in CPO are disjoint unions with elements of different summands incompatible. Usually, one considers *locally continuous* endofunctors on categories of cpos, i. e., endofunctors H where each derived function $\mathsf{CPO}(X, Y) \to \mathsf{CPO}(HX, HY)$ is continuous. Unfortunately, on CPO not every locally continuous functor has a final coalgebra. 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 [12], and moreover, H is iteratable.
(iii) Let CMS be the category of complete metric spaces with distances measured in the interval $[0, 1]$ together with non-expanding maps $f : X \to Y$; i. e., $d_Y(fx, fy) \leq d_X(x, y)$ for all $x, y \in X$. A functor H on CMS is called *contracting*

if there exists a constant $\varepsilon < 1$ such that each derived function $\mathsf{CMS}(X,Y) \to \mathsf{CMS}(HX,HY)$ is a contracting map; i. e., $d_{HX,HY}(Hf,Hg) \leq \varepsilon \cdot d_{X,Y}(f,g)$ for all non-expanding maps $f,g : X \to Y$. Contracting functors are iteratable, see [7].

Remark 2.3. We denote for an 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 [21], the structure map of the final coalgebra is an isomorphism, and consequently, TX is a coproduct of HTX and X with injections

$$\eta^H_X : X \to TX \qquad \text{and} \qquad \tau^H_X : HTX \to TX\,.$$

Again, the superscripts will be dropped if confusion is unlikely.

It has been shown in the previous work [1, 26] that the object assignment T gives rise to a monad on $\mathcal{A}$. And this monad is characterized by a universal property—it is the *free completely iterative monad* on H.

Examples 2.4. (i) Given any signature $\Sigma = (\Sigma_n)_{n<\omega}$ there is an associated polynomial endofunctor $H_\Sigma X = \Sigma_0 + \Sigma_1 \times X + \Sigma_2 \times X^2 + \cdots$ on Set, which is iteratable. Consider the algebra $T_\Sigma X$ of finite and infinite Σ-trees over the set X. That is, (ordered and rooted) trees labelled so that a node with n children, $n > 0$, is labelled by an n-ary operation symbol, and leaves are labelled by constant symbols or elements of X.

Notice that $T_\Sigma X$ is the final coalgebra of $H_\Sigma(\_)+X$. The coalgebra structure is given by the inverse of tree tupling. Finally, notice that T_Σ is a monad on Set.
(ii) A functor $H : \mathsf{Set} \to \mathsf{Set}$ is finitary (i. e., it preserves filtered colimits) iff it is a quotient of some polynomial functor H_Σ, see [8], III.4.3. The latter means that we have a natural transformation $\varepsilon : H_\Sigma \to H$ 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

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

for $\sigma \in \Sigma_n$, $\rho \in \Sigma_m$ and $\sigma(x_1,\ldots,x_n), \rho(y_1,\ldots,y_m) \in H_\Sigma X$ for some set X including all x_i and y_j. Adámek [2] has 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. Example: The functor H which assigns to a set X the set 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.
(iii) Consider the finite power set functor $\mathcal{P}_\mathrm{f} : \mathsf{Set} \to \mathsf{Set}$. Under the Anti-Foundation Axiom (AFA), its final coalgebra is the set HF_1 of hereditarily finite

sets; see [13]. 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 [30]; 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.

(iv) For our later applications we shall find it convenient to work with an iteratable endofunctor $H : \mathsf{Set} \to \mathsf{Set}$ that has a locally continuous lifting H' on CPO; i. e., $U \cdot H' = H \cdot U$ where $U : \mathsf{CPO} \to \mathsf{Set}$ is the forgetful functor. It can be proved analogously as in [8], Theorem IV.5.3, that H' is iteratable. Moreover, for every CPO X, there is a cpo structure on the final coalgebra $\mathcal{T}(H)X$ of $H(\_) + X$ on Set; and this yields the final coalgebra of $H'(\_) + X$ on CPO. It follows that $U \cdot \mathcal{T}(H') = \mathcal{T}(H) \cdot U$. For example, every polynomial functor H_Σ has a locally continuous lifting H', and $\mathcal{T}(H')X$ is the Σ-tree algebra $T_\Sigma X$ with the order induced by the order of the cpo X.

(v) For a set endofunctor H with a contracting lifting H' on CMS; i. e., $U \cdot H' = H \cdot U$ for $U : \mathsf{CMS} \to \mathsf{Set}$ the forgetful functor, we have that H is iteratable and $U \cdot \mathcal{T}(H') = \mathcal{T}(H) \cdot U$. In fact, this follows from the results of [7] since U preserves limits. Any polynomial functor H on Set has a contracting lifting to CMS. For $HX = X^n$, define $H'(X, d) = (X^n, \frac{1}{2} d_{\max})$ (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 metric, see [27] for details.

Example 2.5. 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. That is, a scheme is a set of formal equations

$$f(x_1, \ldots, x_n) \approx t^f(x_1, \ldots, x_n), \quad f \in \Phi_n, \quad n \in \mathbb{N}\,. \tag{2.2}$$

Now a signature is the same as a functor $\mathbb{N} \to \mathsf{Set}$, where $\mathbb{N}$ is understood as the discrete category with natural numbers as objects. Now observe that the names of the variables in (2.2) do not matter. More precisely, any RPS as in (2.2) gives rise to a natural transformation $\Phi \to T_{\Sigma+\Phi} \cdot J$, where $J : \mathbb{N} \to \mathsf{Set}$ is the inclusion functor mapping any number n to the set $\{\, 0, \ldots, n-1 \,\}$. Notice that there is no need to consider only finite signatures Φ here. Moreover, this natural transformation extends the classical notion of RPS in the sense that by taking $T_{\Sigma+\Phi}$ we allow infinite trees on the right-hand sides of systems like (2.2). Here is how we render this example in our approach: Observe that $T_{\Sigma+\Phi} = \mathcal{T}(H_\Sigma + H_\Phi)$ where H_Σ and H_Φ denote the polynomial functors associated to Σ and Φ, respectively. Thus, to give the above natural transformation is the same as to give a natural transformation $H_\Phi \to \mathcal{T}(H_\Sigma + H_\Phi)$.

Example 2.6. 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 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 (1.1) gives a natural transformation $H_\Phi \to \mathcal{T}(H_\Sigma + H_\Phi)$.

In this paper we will 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 $V \to \mathcal{T}(H + V)$ where H, V, and $H + V$ are iteratable endofunctors of the category $\mathcal{A}$.

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. In the classical theory one works with continuous algebras; i. e., algebras carried by a cpo such that all operations are continuous maps. Here we work with *completely iterative algebras*, see [26], and *complete Elgot algebras*, see [5].

Definition 3.1. *Let $H : \mathcal{A} \to \mathcal{A}$ be an endofunctor. By a* flat equation morphism *in an object A (of parameters) we mean a morphism $e : X \to HX + A$. If $a : HA \to A$ is an H-algebra, a* solution *of e in A is a morphism $e^\dagger : X \to A$ such that $e^\dagger = [a, A] \cdot (He^\dagger + A) \cdot e$. We call A a* completely iterative algebra (CIA) *if every flat equation morphism has a unique solution.*

We explain this notion in the classical setting. Let Σ be a signature of givens. A flat recursive system of equations is given by a set X of variables and for each variable $x \in X$ a formal equation

$$x \approx t \quad \text{where } t = \sigma(x_1, \dots, x_n), \sigma \in \Sigma_n, x_i \in X \text{ or } t \in A.$$

The system corresponds to $e : X \to H_\Sigma X + A$, and a solution $e^\dagger$ assigns to every variable an element of A such that the formal equations become identities in A.

Examples 3.2. (i) Final coalgebras. In [26] it is shown that for a final H-coalgebra $\alpha : T \to HT$ the inverse $\tau : HT \to T$ of the structure map yields a CIA. Analogously, for every object X of $\mathcal{A}$ a final coalgebra TX of $H(\_) + X$ yields a CIA, see Theorem 3.9 below. Furthermore, the algebras $T_\Sigma X$ of all Σ-trees over a set X are CIAs, see Example 2.4(i).
(ii) 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 $\mathcal{P}_\mathrm{f}$. This is a special case of the first example.
(iii) Let H be a contracting endofunctor of the category CMS of complete metric spaces, see Example 2.2(iii). Then any non-empty H-algebra is completely iterative; see [5] for this and further examples.

Remark 3.3. In order to define Elgot algebras below we will need two operations. The first one formalizes the renaming of parameters in a flat equation morphism. For a flat equation morphism $e : X \to HX + A$ and a morphism $h : A \to B$ 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 equations morphisms $e : X \to HX + Y$ and $f : Y \to HY + A$ where the parameters of the first are the variables of the second into one "simultaneous" flat equation morphism:

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

Definition 3.4. *A (complete)* Elgot H-algebra *is a triple* $(A, a, (\_)^\dagger)$*, where* (A, a) *is an* H*-algebra, and* $(\_)^\dagger$ *assigns to every flat equation morphism* $e : X \to HX + A$ *a solution* $e^\dagger : X \to A$ *such that the following two laws hold:*
Functoriality: *Solution respects renaming of variables. Given two flat equation morphisms* $e : X \to HX + A$ *and* $f : Y \to HY + A$ *and a morphism* $h : X \to Y$ *of equations between them; i. e.,* $f \cdot h = (Hh + A) \cdot e$*, we then have* $e^\dagger = f^\dagger \cdot h$*.*
CIA-identity. *Simultaneous recursion may be performed sequentially. For every flat equation morphisms* $e : X \to HX + Y$ *and* $f : Y \to HY + A$*, the solution of the combined equation morphism* $f \blacksquare e$ *may be obtained by first solving* f *and then solving* e*, "plugging in"* $f^\dagger$ *for the parameters:* $(f^\dagger \bullet e)^\dagger = (f \blacksquare e)^\dagger \cdot \mathsf{inl}$*.*

Remark 3.5. (i) Notice that there is a notion of (non-complete) Elgot algebra, see [5]. However, since we are only concerned with complete Elgot algebras here, we will henceforth abuse terminology and just speak of Elgot algebras in lieu of complete ones.

(ii) The axioms of Elgot algebras are inspired by the axioms of iteration theories of Bloom and Ésik [15]. In fact, the two laws above are essentially "flat" versions of the functorial dagger implication and the left pairing identity (also known as Bekić-Scott identity) from [15].

One justification for the above axioms is that Elgot algebras turn out to be the Eilenberg-Moore category of the monad T, see Remark 2.3. We shall mention this result at the end of this section. Applied to a polynomial functor H_Σ on Set that means a Σ-algebra A is an Elgot algebra precisely if all Σ-trees over A can be canonically interpreted in A.

Examples 3.6. (i) Completely iterative algebras are Elgot algebras. Cf. [5].
(ii) Continuous algebras. Let H be a locally continuous endofunctor on CPO, see Example 2.2(ii). It was shown in [5] that any H-algebra (A, a) with a least element $\bot$ is an Elgot algebra when to a flat equation morphism $e : X \to HX+A$ the least solution $e^\dagger$ is assigned.
(iii) Suppose that $H : \mathsf{Set} \to \mathsf{Set}$ is a functor with a locally continuous lifting $H' : \mathsf{CPO} \to \mathsf{CPO}$, see Example 2.4(iv). We call an H-algebra $\alpha : HA \to A$ CPO-*enrichable* if there exists a complete partial order $\sqsubseteq$ on A such that A becomes a continuous algebra $\alpha : H'(A, \sqsubseteq) \to (A, \sqsubseteq)$ with a least element.

Any CPO-enrichable algebra A is an Elgot algebra: to every equation morphism $e : X \to HX + A$ assign the least solution of $e : (X, \leq) \to H'(X, \leq) + (A, \sqsubseteq)$, where $\leq$ is the discrete order on X; i. e., $x \leq y$ iff $x = y$.

Definition 3.7. *A* homomorphism h *from an Elgot algebra* $(A, a, (\_)^\dagger)$ *to an Elgot algebra* $(B, b, (\_)^\ddagger)$ *is a morphism* $h : A \to B$ *preserving solutions: for each* $e : X \to HX + A$ *we have* $(h \bullet e)^\ddagger = h \cdot e^\dagger$.

Proposition 3.8. *Every homomorphism* $h : (A, a, (\_)^\dagger) \to (B, b, (\_)^\ddagger)$ *of Elgot algebras is a homomorphism of* H*-algebras; i. e.,* $h \cdot a = b \cdot Hh$. *The converse is false in general. If, however,* A *and* B *are CIAs then any* H*-algebra morphism is a homomorphism of Elgot algebras.*

The classical theory of recursive program schemes rests on the fact that in any continuous algebra A all Σ-trees over A can be interpreted; i. e., there is a canonical map $T_\Sigma A \to A$. 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 [26] and Theorem 5.6 of [5].

Theorem 3.9. *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 of $H(\_)+Y$, the inverse $[\tau_Y, \eta_Y] : HTY + Y \to Y$ of α_Y gives a CIA, which as an Elgot algebra is free on Y. Conversely, given a free Elgot H-algebra $(TY, \tau_Y, (\_)^\dagger)$ with a universal arrow $\eta_Y : Y \to TY$, 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 of $H(\_)+Y$.

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 3.10. *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* $\mathcal{A}$; *in symbols,* $L \dashv U : \mathsf{Alg}^\dagger H \to \mathcal{A}$. *The left-adjoint* L *assigns to each object* Y *of* $\mathcal{A}$ *a free Elgot algebra* $(TY, \tau_Y, (\_)^\dagger)$ *on* Y. *The components of the unit* η *are the universal arrows* $\eta_Y : Y \to TY$ *of the free Elgot algebras. The counit* ε *gives for each Elgot algebra* $(A, a, (\_)^\ddagger)$ *the unique homomorphism* $\widetilde{a} : TA \to A$ *such that* $\widetilde{a} \cdot \eta_A = id$, *and we also have* $\widetilde{a} = (\alpha_A)^\ddagger$ *where the coalgebra structure* $\alpha_A : TA \to HTA + A$ *is considered as a flat equation morphism.*

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_Y is the (unique) solution of α_{TY}, considered as a flat equation morphism with parameters in TY.*

(ii) *A natural transformation $\alpha^H : T \to HT + Id$ expressing the coalgebra structures of TY.*

(iii) *A natural transformation $\tau^H : HT \to T$ such that $[\tau^H, \eta^H]$ is the pointwise inverse of α^H.*

(iv) *A "canonical embedding" κ^H of H into T: $\kappa^H \equiv H \xrightarrow{H\eta^H} HT \xrightarrow{\tau^H} T$.*

As always, we just write $\mathcal{T}(H)$, or even just T, to denote the monad of 3.10(i) above, and we shall frequently drop the superscripts when dealing with the structure coming from a single endofunctor H.

Theorem 3.11. [5] *The category $\mathsf{Alg}^\dagger\, H$ of Elgot algebras is isomorphic to the Eilenberg-Moore category $\mathcal{A}^T$ of monadic T-algebras. The isomorphism assigns to an Elgot algebra $(A, a, (\_)^\ddagger)$ the Eilenberg-Moore algebra $\widetilde{a} : TA \to A$.*

4 Second Order Substitution

In [1, 26] it is proved that the monad $\mathcal{T}(H)$ of Theorem 3.10 is characterized by an important universal property—it is the free *completely iterative monad* on H. In this short abstract we will not need the full strength of this result in order to present our results on recursive program schemes. However, the freeness of $\mathcal{T}(H)$ specializes to second order substitution. We will first recall this concept for Σ-trees, see e.g. [16], and then present a generalization to the final coalgebras $\mathcal{T}(H)$.

Example 4.1. Let Σ and Γ be signatures. 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 $\Sigma \to T_\Gamma \cdot J$ between signatures (considered as functors $\mathbb{N} \to \mathsf{Set}$), and this gives rise to a natural transformation $\lambda : H_\Sigma \to T_\Gamma$. Now for any set X of variables the action of the second order substitution $\overline{\lambda}_X : T_\Sigma X \to T_\Gamma X$ is to replace every Σ-symbol in a tree t from $T_\Sigma X$ by its implementation according to λ; i. e., if $t = \sigma(t_1, \ldots, t_n)$ with $\sigma \in \Sigma(n)$ and if $t'(x_1, \ldots x_n) \in T_\Gamma X$ is the implementation of σ, then $\overline{\lambda}_X(t) = t'(\overline{\lambda}_X(t_1), \ldots, \overline{\lambda}_X(t_n))$. 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 the assignment $x + y \mapsto b(x, u(y))$, $x * y \mapsto b(u(x), y)$, and $1 \mapsto u(c)$. For the set $Z = \{ z, z' \}$, the second order substitution morphism $\overline{\lambda}_Z$ acts for example as follows: $(z + z') * 1 \mapsto b(u(b(z, u(z'))), u(c))$. Notice that although in this example we assigned finite trees to all Σ-symbols in general we may assign also infinite Γ-trees. 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. Finally, the reader may check that $\overline{\lambda}$ is again natural in X and that it is in fact a monad morphism from T_Σ to T_Γ.

Theorem 4.2. [1] *Let H and K be iteratable functors. Suppose that $\lambda : H \to \mathcal{T}(K)$ is an* ideal *natural transformation; i. e., there exists some natural transformation $\lambda' : H \to K\mathcal{T}(K)$ with $\tau^K \cdot \lambda' = \lambda$. Then there exists a unique monad morphism $\overline{\lambda} : \mathcal{T}(H) \to \mathcal{T}(K)$ such that $\overline{\lambda} \cdot \kappa^H = \lambda$.*

Second-order substitution is not trivial to define in the classical setting, and so it is significant that it generalizes to our setting.

5 Uninterpreted Recursive Program Schemes

In the classical treatment of recursive program schemes one gives an uninterpreted semantics to systems as in (2.2) 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 given signature Σ. 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.

We have seen in Example 2.4(i) that Σ-trees can be characterized as the final coalgebra of the polynomial endofunctor associated to Σ. It is the universal property of this final coalgebra which allows one to give a semantics to the given RPS. We will in this section 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 Example 2.5.

Definition 5.1. *Let V and H be endofunctors on $\mathcal{A}$. A* recursive program scheme *(or* RPS*, for short) is a natural transformation $e : V \to \mathcal{T}(H + V)$.*

The RPS e is called guarded *if it factors as follows:*

$$e \equiv V \xrightarrow{f} H\mathcal{T}(H+V) \xrightarrow{\mathsf{inl} * \mathcal{T}(H+V)} (H+V)\mathcal{T}(H+V) \xrightarrow{\tau^{H+V}} \mathcal{T}(H+V)\,,$$

for some natural transformation $f : V \to H\mathcal{T}(H + V)$.

A solution *of e is an ideal transformation $e^\dagger : V \to \mathcal{T}(H)$ such that the following equation holds:*

$$e^\dagger \equiv V \xrightarrow{e} \mathcal{T}(H+V) \xrightarrow{\overline{[\kappa^H, e^\dagger]}} \mathcal{T}(H)\,.$$

Remark 5.2. Recall that $\overline{[\kappa^H, e^\dagger]}$ is the unique monad morphism extending $\sigma = [\kappa^H, e^\dagger] : H + V \to \mathcal{T}(H)$, see Theorem 4.2. Observe that therefore it is important to require that $e^\dagger$ be an ideal transformation since otherwise $\overline{\sigma}$ is not defined.

Remark 5.3. From Example 2.5, 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. Furthermore, any recursive program scheme as in (2.2) which is in Greibach normal form yields a guarded recursive program scheme in the sense of Definition 5.1.

Theorem 5.4. *Any guarded recursive program scheme has a unique solution.*

Sketch of Proof. Let $H : \mathcal{A} \to \mathcal{A}$ be any iteratable functor. Then $T = \mathcal{T}(H)$ is a final coalgebra of the functor $H \cdot \_ + Id$ on the endofunctor category $[\mathcal{A}, \mathcal{A}]$. This result extends to the comma-category $H/\mathbf{Mon}(\mathcal{A})$ whose objects are pairs $(S, \sigma : H \to S)$ where S is a monad on $\mathcal{A}$ and σ is a natural transformation, and whose morphisms $h : (S_1, \sigma_1) \to (S_2, \sigma_2)$ are monad morphisms $h : S_1 \to S_2$ with $h \cdot \sigma_1 = \sigma_2$. In fact, we obtain a functor $\mathcal{H}$ on $H/\mathbf{Mon}(\mathcal{A})$ with $\mathcal{H}(S, \sigma) = (HS + Id, \mathsf{inl} \cdot H\eta)$, where $\eta : Id \to S$ is the unit of the monad S. Furthermore, T together with $\kappa^H : H \to T$, see Theorem 3.10(iv), is the final $\mathcal{H}$-coalgebra.

Now suppose we are given a guarded RPS $e : V \to \mathcal{T}(H+V)$. Then we have a natural transformation $\sigma = \mathsf{inl} \cdot [H\eta^{H+V}, f] : H + V \to H\mathcal{T}(H+V) + Id$ 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 in the sense of [1]. Use the full universal property of the free completely iterative monad $\mathcal{T}(H+V)$ to obtain a monad morphism $\overline{e} : \mathcal{T}(H+V) \to H\mathcal{T}(H+V) + Id$, see [1], Theorem 4.14. 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 the full version [27] for details.

Remark 5.5. (i) The first part of the proof of Theorem 5.4 showing the finality of $\mathcal{T}(H)$ and defining the monad morphism h uses similar ideas than the proof of the main result of [19]. However, the second part in which it is proved that $e^\dagger$ is a unique solution of e is new. It connects solutions to the (generalized) second order substitution as presented in Theorem 4.2.

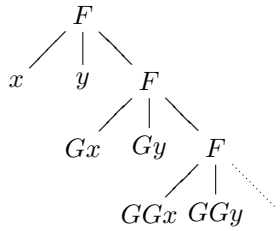
(ii) Notice that in the classical setting not every recursive program scheme which has a solution needs to be in Greibach normal form. For example, consider the system formed by the first equation in (1.1) and by the equation $\psi(x) \approx \varphi(\psi(x))$. This system gives rise to an unguarded RPS. Thus, Theorem 5.4 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 5.4.

Example 5.6. 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 one wants to define a commutative binary operation φ by the formal equation

$$\varphi(x,y) \approx F(x,y,\varphi(Gx,Gy))\,. \tag{5.3}$$

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 (5.3) gives rise to a guarded RPS $e : V \to \mathcal{T}(H+V)$. In fact, to see the naturality use the description of the terminal coalgebra $\mathcal{T}(H+V)Y$ given in [2], see Example 2.4(ii). 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—any solution will be commutative. In fact, the components of the uninterpreted solution $e^\dagger_X : VX \to \mathcal{T}(H)X$ assign to an unordered pair $\{\, x, y \,\}$ in VX the tree



where for every node labelled by F the first two children are unordered.

6 Interpreted Recursive Program Schemes

We have seen in the previous section that for any 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 Elgot algebra $(A, a, (\_)^*)$ admits an interpreted solution of any guarded recursive program scheme. Moreover, if A is a CIA, interpreted solutions are unique. We also state that uninterpreted solutions and interpreted ones correspond to one another. This is a fundamental result for algebraic semantics.

Definition 6.1. *Let $(A, a, (\_)^*)$ be an Elgot algebra w.r.t. H and let $e : V \to \mathcal{T}(H+V)$ be an RPS. An* interpreted solution *of e in A is a structure of a*

V-algebra $e^\dagger_A : VA \to A$, such that the $(H+V)$-algebra $[a, e^\dagger_A] : (H+V)A \to A$ is an Elgot algebra and such that the equation

$$e^\dagger_A \equiv VA \xrightarrow{e_A} \mathcal{T}(H+V)A \xrightarrow{\widetilde{[a,e^\dagger_A]}} A \tag{6.4}$$

holds, where the second arrow denotes the Eilenberg-Moore algebra structure associated to the Elgot algebra $[a, e^\dagger_A]$, see Theorem 3.11.

Theorem 6.2. *Let $(A, a, (\_)^*)$ be an Elgot algebra w. r. t. H and let $e : V \to \mathcal{T}(H+V)$ 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.

Sketch of Proof. Recall the $\mathcal{H}$-coalgebra structure $\overline{e}$ from the proof of Theorem 5.4. Let us write T for $\mathcal{T}(H+V)$. Then the component at A of $\overline{e}$ yields a flat equation morphism $\overline{e}_A : TA \to HTA + A$ w.r.t. the given Elgot algebra. Denote its solution $(\overline{e}_A)^*$ by β, and define

$$e^\dagger_A \equiv VA \xrightarrow{\mathsf{inr}} (H+V)A \xrightarrow{\kappa^{H+V}_A} \mathcal{T}(H+V)A \xrightarrow{\beta} A\,.$$

A non-trivial proof shows that this morphism is an interpreted solution of e, and that it is the unique solution if A is a CIA. For details see [27].

Finally, we state the "Fundamental Theorem of Algebraic Semantics", which establishes that uninterpreted and interpreted solutions are connected in the "proper way".

Theorem 6.3. *Let $(A, a, (\_)^*)$ be an Elgot algebra considered as an Eilenberg-Moore algebra $\widetilde{a} : \mathcal{T}(H)A \to A$, and let e be any guarded recursive program scheme. If $e^\dagger_A : VA \to A$ is the interpreted solution of e in A of Theorem 6.2 and $e^\dagger : V \to \mathcal{T}(H)$ is the (uninterpreted) solution of Theorem 5.4, then the equation $\widetilde{a} \cdot (e^\dagger)_A = e^\dagger_A$ holds.*

Remark. Notice that $(e^\dagger)_A$ is the component at A of the natural transformation $e^\dagger$ whereas $e^\dagger_A$ is not a component of some natural transformation but merely a morphism from VA to A.

Sketch of Proof. Recall the morphisms $h : \mathcal{T}(H+V) \to \mathcal{T}(H)$ and $\beta : \mathcal{T}(H+V)A \to A$ from the proofs of Theorems 5.4 and 6.2. It is not difficult to show that the equation $\beta = \widetilde{a} \cdot h_A$ holds. Precompose both sides with $\kappa^{H+V}_A \cdot \mathsf{inr} : VA \to \mathcal{T}(H+V)A$ to obtain the desired result.

6.1 Interpreted Solutions in CPOs

We shall show in this subsection that if we have $\mathcal{A} = \mathsf{CPO}$ as our base category, then interpreted solutions of a guarded RPS e in an Elgot algebra $(A, a, (\_)^*)$ are given as least fixed points of a continuous function on a function space. In this way we recover denotational semantics from our categorical interpreted semantics of recursive program schemes.

Example 6.4. The standard example in classical algebraic semantics is the factorial function. So let Σ be a signature containing a constant zero, two unary symbols succ and pred, a binary symbol $*$ and a ternary one if. The interpretation we have in mind is the natural numbers. The signature Φ of the recursively defined operations consists just of one unary symbol f. Consider the formal recursive equation

$$f(n) \approx \mathsf{if}(n, \mathsf{succ}(\mathsf{zero}), f(\mathsf{pred}(n)) * n)) \tag{6.5}$$

defining the factorial function. It gives rise to a guarded recursive program scheme $e : H_\Phi \to \mathcal{T}(H_\Sigma + H_\Phi)$ as demonstrated in Example 2.5.

To obtain a suitable Elgot algebra in which we can find an interpreted solution of e, we turn the natural numbers into a CPO. Let $\mathbb{N}_\perp$ be the flat CPO obtained from the discretely ordered set of natural numbers by adding a least element $\perp$. We equip $\mathbb{N}_\perp$ with the obvious continuous operations corresponding to the names of the operation symbols of Σ. Hence we have a continuous Σ-algebra, and therefore $\mathbb{N}_\perp$ is an Elgot H_Σ-algebra, as in Example 3.6(iii).

The interpreted solution $e^\dagger_{\mathbb{N}_\perp} : H_\Phi \mathbb{N}_\perp \to \mathbb{N}_\perp$ is given by a function on $\mathbb{N}_\perp$. But how do we know that this function is the desired factorial function? Usually one would simply regard the RPS (6.5) itself as a continuous function R which maps every continuous operation f on $\mathbb{N}_\perp$ to $\mathsf{if}_{\mathbb{N}_\perp}(\_, 1, f(\mathsf{pred}_{\mathbb{N}_\perp}(\_) *_{\mathbb{N}_\perp} \_)$; i. e., we interpret all the operation symbols of Σ in the algebra $\mathbb{N}_\perp$. It is clear that the least fixed point of R is the desired factorial function. And we show that this least fixed point coincides with the interpreted solution obtained from Theorem 6.2.

In general any recursive program scheme can be turned into a continuous function R on the function space $\mathsf{CPO}(VA, A)$ and the least fixed point of R is the same as the interpreted solution obtained from Theorem 6.2.

We assume throughout this subsection that H and V are locally continuous (and, as always, iteratable) endofunctors of CPO. We also consider a fixed guarded RPS $e : V \to \mathcal{T}(H+V)$, and an H-algebra (A, a) with a least element $\perp$. By Example 3.6(ii), we know that this carries the structure of an Elgot algebra $(A, a, (\_)^*)$, where $(\_)^*$ assigns to every flat equation morphism a least solution. Furthermore, for any continuous map $f : VA \to A$ we have an Elgot algebra on A with structure $[a, f] : (H + V)A \to A$. Its associated Eilenberg-Moore algebra structure is denoted by $\widetilde{[a, f]}$, see Theorem 3.11.

Theorem 6.5. *The following function R on $\mathsf{CPO}(VA, A)$*

$$f \mapsto VA \xrightarrow{e_A} \mathcal{T}(H + V)A \xrightarrow{\widetilde{[a,f]}} A \tag{6.6}$$

is continuous. Its least fixed point is the interpreted solution $e^\dagger_A : VA \to A$ of Theorem 6.2.

Sketch of Proof. To see the continuity of R is suffices to prove that $\widetilde{(\_)} : \mathsf{CPO}(HA, A) \to \mathsf{CPO}(\mathcal{T}(H)A, A)$ is continuous. Let us write T for $\mathcal{T}(H)$. Recall

from Theorem 3.10 that for any continuous map $a : HA \to A$ the Eilenberg-Moore algebra structure $\widetilde{a}$ is the least solution of the flat equation morphism $\alpha_A : TA \to HTA + A$, i. e., $\widetilde{a}$ is the least fixed point of the continuous function $F(a, -) : \mathsf{CPO}(TA, A) \to \mathsf{CPO}(TA, A)$ 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 $\mathsf{CPO}(HA, A) \to \mathsf{CPO}(TA, A)$. But this is precisely the map $\widetilde{(\_)}$.

We prove that $e^\ddagger_A$ 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 $t_0 = \mathsf{const}_\bot$ and $t_{i+1} = \widetilde{[a, t_i]} \cdot e_A$, for $i \in \mathbb{N}$.

Furthermore, recall that the interpreted solution $e^\ddagger_A$ is defined by $\beta \cdot \kappa_A^{H+V} \cdot \mathsf{inr}$, 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 Theorem 6.2. By Example 3.6(ii), the solution β of g is the join of the chain given by $\beta_0 = \mathsf{const}_\bot$ and $\beta_{i+1} = [a, A] \cdot H(\beta_i + A) \cdot g$, for $i \in \mathbb{N}$.

Observe that $e^\ddagger_A$ is a fixed point of R, see (6.4). Thus, we have $t \sqsubseteq e^\ddagger_A$. 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 [27]. This implies that $\beta \sqsubseteq \widetilde{[a, t]}$ and therefore $e^\ddagger_A = \beta \cdot \kappa_A^{H+V} \cdot \mathsf{inr} \sqsubseteq \widetilde{[a, t]} \cdot \kappa_A^{H+V} \cdot \mathsf{inr} = t$.

Remark 6.6. Suppose that H, V and $H + V$ are iteratable endofunctors of Set, which have locally continuous liftings H', V' and $H' + V'$ to CPO. Then we have $\mathcal{T}(H + V) \cdot U = U \cdot \mathcal{T}(H' + V')$, see Example 2.4(iv). Furthermore, assume that the guarded RPS $e : V \to \mathcal{T}(H + V)$ has a lifting $e' : V' \to \mathcal{T}(H' + V')$; 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 3.6(iii). Then we can apply Theorem 6.5 to obtain an interpreted solution $e^\ddagger_A$ of e in the algebra A as a least fixed point of the above function R of (6.6).

Example 6.7. (i) Suppose we have signatures Σ and Φ. Then the polynomial functors H_Σ and H_Φ satisfy the requirements of Remark 6.6. Consider any system as in (2.2) in Greibach normal form, and form the associated guarded RPS $e : H_\Phi \to \mathcal{T}(H_\Sigma + H_\Phi)$. Then e has a lifting e'. 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 which assigns to any continuous algebra structure $\varphi : H_\Phi A \to A$ the algebra structure $\widetilde{[a, \varphi]} \cdot e'_A$. It maps for a given continuous Φ-algebra structure φ on A any operation $f_A : A^n \to A$, $f \in \Phi_n$, to the operation that computes the right-hand side t^f_A, where operation symbols of Σ are interpreted according to a and symbols of Φ according to the given φ.

Theorem 6.5 states that an interpreted solution $e^\ddagger_A$ of e in the algebra A is 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 Example 6.4. Then Theorem 6.5 states that the interpreted solution of the RPS (6.5) in the Elgot algebra $\mathbb{N}_\bot$ is

obtained as the least fixed point of the function R of Example 6.4. That is, the interpreted solution gives the desired factorial function.
(iii) Recall the guarded RPS e from Example 5.6 and its uninterpreted solution. Consider again the algebra $\mathbb{N}_\perp$ together with the following two operations:

$$F_{\mathbb{N}_\perp}(x,y,z) = \begin{cases} x & \text{if } x = y \\ z & \text{else} \end{cases} \qquad G_{\mathbb{N}_\perp}(x) = \begin{cases} \lfloor \frac{x}{2} \rfloor & \text{if } x \in \mathbb{N} \\ \perp & x = \perp \end{cases}$$

Since the first operation obviously satisfies $F_{\mathbb{N}_\perp}(x,y,z) = F_{\mathbb{N}_\perp}(y,x,z)$ 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 $\mathbb{N}_\perp$ is a continuous H'-algebra. In fact, the existence of the lifting H' follows from the fact that the unordered pair functor $V : \mathsf{Set} \to \mathsf{Set}$ 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 w.r.t. $H : \mathsf{Set} \to \mathsf{Set}$, see Example 3.6(iii). The interpreted solution $e^\ddagger_{\mathbb{N}_\perp} : V\mathbb{N}_\perp \to \mathbb{N}_\perp$ is given by one commutative binary operation $\varphi_{\mathbb{N}_\perp}$ on $\mathbb{N}_\perp$. We leave it to the reader to verify that for natural numbers x and y, $\varphi_{\mathbb{N}_\perp}(x,y)$ is the natural number represented by the greatest common prefix in the binary representation of x and y, e. g., $\varphi_{\mathbb{N}_\perp}(12,13) = 6$. Notice that we do not have to prove separately that $\varphi_{\mathbb{N}_\perp}$ is commutative. The way we have formed the RPS e in Example 5.6 ensures that the interpreted solution will 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 $f : A \to A$ 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 $H' : \mathsf{CPO} \to \mathsf{CPO}$ 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 $e : H_\Phi \to \mathcal{T}(H_\Sigma + H_\Phi)$, see Example 2.6. The RPS e has a lifting $e' : V' \to \mathcal{T}(H' + V')$, where V' denotes the lifting of H_Φ. The interpreted solution $e^\ddagger_A : V'A \to A$ gives two continuous functions $\varphi^\dagger, \psi^\dagger : A \to A$. Clearly, we have $\varphi^\dagger(a) = \bigvee_{n \in \mathbb{N}} f^n(a)$, and in particular $\varphi^\dagger(\perp)$ is the least fixed point of f.

6.2 Interpreted Solutions in CMS

In [14] the interpreted semantics of classical recursive program schemes is studied in certain interpretations arising from the use of complete metric spaces. It is proved there that recursive program schemes which are in Greibach normal form admit a unique interpreted solution in any contraction theory. We shall prove a similar result in our categorical setting now.

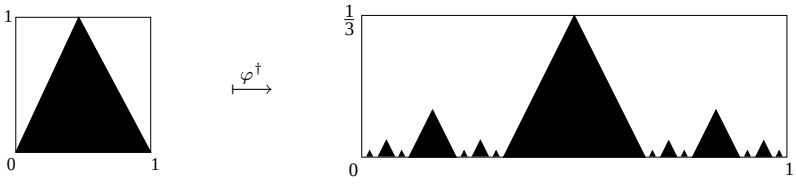
Recall the category CMS of complete metric spaces from Example 3.2(iii), and let H and V be contracting endofunctors. We shall show in this subsection that for any guarded RPS e we can find a unique interpreted solution in any non-empty H-algebra A. More precisely, assume that we have a guarded RPS e :

$V \to \mathcal{T}(H+V)$, and let (A, a) be a non-empty H-algebra. Then A is a CIA, and therefore it has the structure of an Elgot algebra, see Examples 3.2(iii) and 3.6(i). Notice that for any non-expanding map $f : VA \to A$ we obtain an algebra structure $[a, f] : (H+V)A \to A$, thus we have an Eilenberg–Moore algebra $\widetilde{[a, f]} : \mathcal{T}(H+V)A \to A$. As in CPO, the RPS e induces a function R on $\mathsf{CMS}(VA, A)$. 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 6.2. 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 [7].

Corollary 6.8. *The interpreted solution $e^\ddagger_A : VA \to A$ of e in A as obtained in Theorem 6.2 is the unique fixed point of the function R defined as in (6.6).*

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 6.2.

Example 6.9. (i) Self similar sets are solutions of interpreted program schemes. Let (X, d) be a metric space. Then $(C(X), h)$ is the complete metric space formed by all non-empty compact subsets of X with the Hausdorff metric; i. e., for two compact subspaces A and B of X we have $h(A, B) = \max\{\, d(A \to B), d(B \to A)\,\}$, where $d(A \to B) = \max_{a\in A} \min_{b\in B} d(a, b)$. It is well-known that any contracting map f on X gives rise to a contracting map $f' : A \mapsto f[A]$ on $C(X)$, see [11]. 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 $\alpha_A : C(A)^3 \to C(A)$ with $\alpha_A(U, W, Z) = f[U] \cup g[W] \cup h[Z]$ 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 on Set, viz. $e : Id \to \mathcal{T}(H_\Sigma + Id)$, where the identity functor expresses the operation φ. Consider the lifting of the identity to CMS given by $V'(X, d) = (X, \frac{1}{3}d)$. Using this lift, e gives rise to a natural transformation $e' : V' \to \mathcal{T}(H' + V')$. In fact, it is easy to see that all components e'_X are non-expanding maps once the metric on $\mathcal{T}(H' + V')X$ is understood, see the full version [27] or [7] for more details on this. The interpreted solution of e' in the algebra $(C(A), \alpha_A)$ is given by a map $\varphi^\dagger : C(A) \to C(A)$ which maps a non-empty compact subspace U of A to a space of the following form: $\varphi^\dagger(U)$ 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 $\varphi^\dagger(U)$ scaled by $\frac{1}{3}$. For example we have the assignment



(ii) The Cantor "middle third" set c is the unique nonempty subset of the interval $[0,1]$ which satisfies $c = \frac{1}{3}c \cup (\frac{2}{3} + \frac{1}{3}c)$. So as in (i), there is an RPS e on CMS whose interpreted solution in the algebra $C(A)$, $A = [0,1]$, is the Cantor set. More detailed, the functions $f(x) = \frac{1}{3}x$ and $g(x) = \frac{2}{3} + \frac{1}{3}x$ yield a $\frac{1}{3}$-contraction $\alpha_A : C(A)^2 \to C(A)$ with $\alpha(U,W) = f[U] \cup g[W]$, whence an algebra of the functor given by $H'(X,d) = (X^2, \frac{1}{3}d_{\max})$. The formal equation $c \approx \alpha(c,c)$ gives rise to a guarded RPS $e' : V' \to \mathcal{T}(H' + V')$ where V' is the constant functor whose value is the trivial one point space 1, a lifting of the set functor expressing the constant c. Its interpreted solution is given by the function $1 \to C(A)$ choosing the Cantor space.

6.3 Interpreted Solutions in non-well founded sets

In this subsection we show that in hypersets unique solutions of recursive equations as discussed in [13] can be obtained as special instances of our Theorem 6.2. For example, assuming the Anti-Foundation-Axiom there exists a unique set c such that $c = \{\, p(c) \,\}$ where $p(c)$ denotes the finite power set of c, i. e., c uniquely solves the formal equation $x \approx \{\, p(x) \,\}$, where we understand p and the set brackets as formal operation symbols.

Now in order to see how to obtain this and many other examples from our results we shall work with the finite power set functor $\mathcal{P}_{\mathrm{f}} : \mathsf{Set} \to \mathsf{Set}$ whose initial CIA (equivalently, final coalgebra) is HF_1 with a structure map $\tau : \mathcal{P}_{\mathrm{f}}(HF_1) \to HF_1$, see Example 3.2(ii). In the first step we extend this CIA by additional operations needed for our examples. So consider the unary operation $a \mapsto p(a)$ and the binary ones $(a,b) \mapsto a \cup b$ and $(a,b) \mapsto p(a) \times b$ on HF_1. If we describe the type of these operations by the set functor $KX = X + X \times X + X \times X$, then HF_1 is a K-algebra with a structure $b : K(HF_1) \to HF_1$.

Lemma 6.10. *The algebra HF_1 with the structure $k = \tau \cdot \mathcal{P}_{\mathrm{f}} b : \mathcal{P}_{\mathrm{f}} K(HF_1) \to HF_1$ is a CIA of the functor $\mathcal{P}_{\mathrm{f}} K$.*

In fact, this follows from a general result from the theory of CIAs, see the full version [27]. Here we do not have the space to present this in full generality. However, let us point out that from the general result it follows that formal equations with additional operations can be uniquely solved in HF_1 as soon as these additional operations "are guarded" by a set bracket and they "commute with substitutions".

Example 6.11. (i) The formal equation $x \approx \{\, p(x) \,\}$ gives rise to a flat equation morphism $X \to \mathcal{P}_{\mathrm{f}} KX + HF_1$, where X is a singleton set. Its unique solution $e^\dagger : X \to HF_1$ is essentially a set c with the desired property. Notice that for this example we do not need our machinery for recursive program schemes since no recursive function definitions are made.
(ii) Consider the formal equation $f(x) \approx \{\, f(x) \cup x \,\}$. Take the functor $V = Id$ on Set describing the unary operation f. Then the above formal equation gives rise to a recursive program scheme $e : V \to \mathcal{T}(\mathcal{P}_{\mathrm{f}} K + V)$ analogously as in

Example 2.5. The unique interpreted solution $e^{\ddagger}_{HF_1}$ obtained by Theorem 6.2 is given by a function $f^{\dagger}$ on HF_1 which satisfies $f^{\dagger}(a) = \{ f^{\dagger}(a) \cup a \}$.
(iii) Consider the system of formal equations $f(x,y) \approx \{ p(x) \times f(gx, gy) \}, g(x) \approx \{ x \cup f(x,x) \}$. Describe the operations f and g by the set functor $VX = X \times X + X$. Then as before the given system gives rise to a guarded RPS $e : V \to \mathcal{T}(\mathcal{P}_{\mathrm{f}} K + V)$. And the interpreted solutions $e^{\ddagger}_{HF_1}$ yields two functions $f^{\dagger} : HF_1^2 \to HF_1$ and $g^{\dagger} : HF_1 \to HF_1$ turning the formal equations into valid identities as desired.

7 Conclusions and Future Research

We have presented a general and conceptually clear way of treating the uninterpreted and the interpreted semantics of recursive program schemes. For this we have used recent results on Elgot algebras and from the theory of coalgebras. 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 [25] 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 [29] 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 [20]). It would be good to revisit this part of the classical theory to see whether Elgot algebras suggest tractable classes of interpretations.

References

1. 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.
2. J. Adámek, On a Description of Terminal Coalgebras and Iterative Theories, *Electron. Notes Theor. Comput. Sci.*, 82.1 (2003).
3. J. Adámek, S. Milius and J. Velebil, Free Iterative Theories: A Coalgebraic View, *Math. Structures Comput. Sci.*, 13 (2003), 259–320.
4. J. Adámek, S. Milius and J. Velebil, From Iterative Algebras to Iterative Theories, *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`.
5. J. Adámek, S. Milius and J. Velebil, Elgot Algebras, to appear in *Electron. Notes Theor. Comput. Sci.*, available at the URL `http://www.iti.cs.tu-bs.de/~milius`.
6. J. Adámek and H. E. Porst, On tree coalgebras and coalgebra presentations, *Theoret. Comput. Sci.*, 311 (2004), 257–283.

7. J. Adámek and J. Reitermann, Banach's Fixed-Point Theorem as a Base for Data-Type Equations, *Appl. Categ. Structures*, 2 (1994), 77–90.
8. J. Adámek and V. Trnková, *Automata and Algebras in Categories*. Kluwer Academic Publishers, 1990.
9. P. America and J. Rutten, Solving Reflexive Domain Equations in a Category of Complete Metric Spaces, *J. Comput. System Sci.*, 39 (1989), 343–375.
10. A. Arnold and M. Nivat, The metric space of infinite trees. Algebraic and topological properties, *Fund. Inform.* III, no. 4 (1980), 445–476.
11. M. F. Barnsley, *Fractals everywhere*, Academic Press 1988.
12. M. Barr, Terminal coalgebras in well-founded set theory, *Theoret. Comput. Sci.*, 114 (1993), 299–315.
13. J. Barwise and L. S. Moss, *Vicious Circles*, CSLI Publications, Stanford, 1996.
14. 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.
15. S. L. Bloom and Z. Ésik, *Iteration Theories: The equational logic of iterative processes,* EATCS Monographs on Theoretical Computer Science, Berlin: Springer-Verlag (1993).
16. B. Courcelle, Fundamental properties of infinite trees, *Theoret. Comput. Sci.*, 25 (1983), no. 2, 95–169.
17. 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.
18. C. C. Elgot, S. L. Bloom and R. Tindell, On the Algebraic Structure of Rooted Trees, *J. Comput. System Sci.*, 16 (1978), 361–399.
19. N. Ghani, C. Lüth and F. De Marchi, Solving Algebraic Equations using Coalgebra, *Theor. Inform. Appl.*, 37 (2003), 301–314.
20. I. Guessarian, *Algebraic Semantics*. Lecture Notes in Comput. Sci., 99, Springer, 1981.
21. J. Lambek, A Fixpoint Theorem for Complete Categories, *Math. Z.*, 103 (1968), 151–161.
22. T. Leinster, General self-similarity: an overview, e-print math.DS/0411343 v1.
23. T. Leinster, A general theory of self-similarity I, e-print math.DS/041344.
24. T. Leinster, A general theory of self-similarity II, e-print math.DS/0411345.
25. 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).
26. S. Milius, Completely Iterative Algebras and Completely Iterative Monads, *Inform. and Comput.*, 196 (2005), 1–41.
27. S. Milius and L. S. Moss, The Category Theoretic Solution of Recursive Program Schemes, full version, available at the URL `http://www.iti.cs.tu-bs.de/~milius`.
28. L. S. Moss, Parametric Corecursion, *Theoret. Comput. Sci.*, 260 (2001), no. 1–2, 139–163.
29. L. S. Moss, The Coalgebraic Treatment of Second-Order Substitution and Uninterpreted Recursive Program Schemes, preprint, 2002.
30. J. Worrell, On the Final Sequence of a Finitary Set Functor, accepted for publication in *Theoret. Comput. Sci.*