Theoretical Computer Science 412 (2011) 4969–4988

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

On second-order iterative monads

Jiří Adámek [a], Stefan Milius [a,*], Jiří Velebil [b,1]

[a] Institut für Theoretische Informatik, Technische Universität Braunschweig, Germany
[b] Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic

ARTICLE INFO

Keywords:
Algebraic trees
Recursive program schemes
Ideal theory
Monads

ABSTRACT

B. Courcelle studied algebraic trees as precisely the solutions of all recursive program schemes for a given signature in Set. He proved that the corresponding monad is iterative. We generalize this to recursive program schemes over a given finitary endofunctor H of a "suitable" category. A monad is called second-order iterative if every guarded recursive program scheme has a unique solution in it. We construct two second-order iterative monads: one, called the second-order rational monad, S^H, is proved to be the initial second-order iterative monad. The other one, called the context-free monad, C^H, is a quotient of S^H and in the original case of a polynomial endofunctor H of Set we prove that C^H is the monad studied by B. Courcelle. The question whether these two monads are equal is left open.

1. Introduction

Recursive program schemes formalize the construction of new programs from the given ones by solving a recursive system of second-order equations. Building on the classical work of Bruno Courcelle [11] we introduce, for every finitary endofunctor H of a locally finitely presentable category, the *context-free monad* C^H of H. In the case where H is the polynomial endofunctor of a signature Σ in Set we prove that C^H is Courcelle's monad of *algebraic trees*, i.e., those Σ-trees that are solutions of recursive program schemes. This monad C^H is a quotient monad of the *second-order rational monad* S^H defined as the colimit of the diagram of all recursive program schemes. This is analogous to our previous construction of the rational monad R^H characterizing solutions of first-order recursive equations of type H; see [4]. In the case of a polynomial functor $H = H_\Sigma$ on Set the monad R^H is given by all rational Σ-trees, i.e., Σ-trees having (up to isomorphism) only a finite set of subtrees; see [19].

Recall from [11] the language $L(t)$ associated to every tree t: this is the language consisting of all words $n_1 \dots n_k f$ where $n_1 \dots n_k$ is a word over ω denoting a path from the root of t to a node (of depth k), and $f \in \Sigma$ is the label of that node. The tree t is rational iff $L(t)$ is a regular language, and, as proved in [13], the tree t is algebraic iff $L(t)$ is a deterministic context-free language. For this reason we call C^H the context-free monad for H.

This paper is part of our research program to provide a new and conceptionally clear approach to algebraic semantics (see e.g. [11,20]) which is a topic at the heart of theoretical computer science. In this new approach we use category theoretic methods and tools instead of general algebraic ones. So in lieu of sets, signatures and trees we consider categories, functors, final coalgebras and monads formed by them. Algebraic trees for a signature are a very important concept in classical

* Corresponding author. Tel.: +49 531 391 9524.
E-mail address: mail@stefan-milius.eu (S. Milius).

1 The third author was supported by the grant MSM 6840770014 of the Ministry of Education of the Czech Republic.

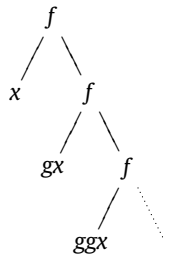
0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.04.027

algebraic semantics providing a semantic domain for uninterpreted solutions of recursive program schemes. So it is an important question how algebraic trees can be captured in our more general category theoretic approach to algebraic semantics, and we present the context-free monad as an answer. In addition, the move from signatures to endofunctors also allows to extend the notion of recursive program scheme. So in our new semantics we can capture recursive equations that the classical work cannot deal with; for example, equational laws between given operations of a program scheme may be considered directly in our approach—this is discussed in [24].

Let us now explain the results of this paper a bit more in detail. Recall that a *recursive program scheme* (or *rps* for short) defines new operations $\varphi_1, \ldots, \varphi_k$ of given arities $n_1, \ldots, n_k$ recursively, using given operations represented by symbols from a signature Σ. An rps is *guarded* if the right-hand sides of the equations have the leading symbol in Σ. Here is an example:

$$\varphi(x) = f(x, \varphi(gx)) \tag{1.1}$$

is a recursive program scheme defining a unary operation φ from the givens in $\Sigma = \{f, g\}$ with f binary and g unary. Here we are interested in the so-called uninterpreted semantics, which treats a recursive program scheme as a purely syntactic construct, and so its solution is given by Σ-trees over the given variables. For example, the uninterpreted solution of φ above is the Σ-tree



(1.2)

(here we simply put the terms x, gx, ggx, etc. for the corresponding subtrees).

Observe that if $\Phi = \{\varphi_1, \ldots, \varphi_k\}$ denotes the finite signature of the newly defined operations of arities n_i and

$$H_\Phi X = X^{n_1} + \cdots + X^{n_k}$$

is the corresponding polynomial endofunctor of Set, then algebras for H_Φ are just the classical general algebras for the signature Φ. We denote by F^H the free monad on H, thus F^{H_Φ} is the monad of finite Φ-trees. A recursive program scheme can be formalized as a natural transformation

$$e : H_\Phi \to F^{H_\Sigma + H_\Phi}.$$

In fact, $F^{H_\Sigma + H_\Phi}$ is the monad of all finite $(\Sigma + \Phi)$-trees. Since X^{n_i} is a functor representable by n_i, a natural transformation from X^{n_i} into $F^{H_\Sigma + H_\Phi}$ is, by the Yoneda Lemma, precisely an element of $F^{H_\Sigma + H_\Phi}(n_i)$, i.e., a finite $(\Sigma + \Phi)$-tree on n_i variables. Thus, to give a natural transformation e as above means precisely to give k equations, one for each operation symbol φ_i from Φ,

$$\varphi_i(x_0, \ldots, x_{n-1}) = t_i \quad (i = 1, \ldots, k) \tag{1.3}$$

where t_i is a $(\Sigma + \Phi)$-term on $\{x_0, \ldots, x_{n-1}\}$. This is the definition of a recursive program scheme used in [11].

An uninterpreted solution of $e : H_\Phi \to F^{H_\Sigma + H_\varphi}$ is a k-tuple of Σ-trees $t_1^\dagger, \ldots, t_k^\dagger$ such that the above formal equations (1.3) become identities under the simultaneous second-order substitution[2] of t_i for f_i, for $i = 1, \ldots, k$. For example, the tree $t^\dagger(x)$ from (1.2) satisfies the corresponding equality of trees

$$t^\dagger(x) = g(x, t^\dagger(fx)).$$

This concept of solutions was formalized in [24] by means of the free completely iterative monad T^H on a functor H; in the case $H = H_\Sigma$ this is the monad of all (finite and infinite) Σ-trees. We recall this in Section 2. The uninterpreted solution is a natural transformation $e^\dagger : H_\Phi \to T^{H_\Sigma}$ and this leads us to the following reformulation of the concept of an algebraic tree of Courcelle [11]:

A Σ-tree is called *algebraic* if there exists a recursive program scheme (1.3) such that $t = t_1^\dagger$. (Every rational tree is algebraic, and (1.2) shows an algebraic tree that is not rational.)

Courcelle proved that the monad C^{H_Σ} of all algebraic Σ-trees as a submonad of T^{H_Σ} is iterative in the sense of Calvin Elgot [12]. Furthermore, algebraic trees are closed under second-order substitution. In this paper we study, for general finitary endofunctors H, solutions of recursive program schemes in an arbitrary H-pointed monad, i.e., a monad B together with a natural transformation from H to B.

[2] Recall that, in general, a simultaneous second-order substitution replaces in a tree over a signature Γ all operation symbols by trees over another signature; see [11] for classical second-order substitution or [24] for a category theoretic description.