Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 325 (2016) 147–168

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Complete Elgot Monads and Coalgebraic Resumptions†

Sergey Goncharov[1,⋆] Stefan Milius[2] Christoph Rauch[3,⋆]

Lehrstuhl für Theoretische Informatik, Friedrich-Alexander Universität Erlangen-Nürnberg, Germany

Abstract

Monads are used to abstractly model a wide range of computational effects such as nondeterminism, statefulness, and exceptions. *Complete Elgot monads* are monads that are equipped with a *(uniform) iteration operator* satisfying a set of natural axioms, which allows to model *iterative computations* just as abstractly. It has been shown recently that extending complete Elgot monads with free effects (e.g. operations of sending/receiving messages over channels) canonically leads to *generalized coalgebraic resumption monads*, which were previously used as semantic domains for non-wellfounded guarded processes. In this paper, we continue the study of complete Elgot monads and their relationship with generalized coalgebraic resumption monads. We give a characterization of the Eilenberg-Moore algebras of the latter. In fact, we work more generally with Uustalu's *parametrized monads*; we introduce complete Elgot algebras for a parametrized monad and we prove that they form an Eilenberg-Moore category. This is further used for establishing a characterization of complete Elgot monads as those monads whose algebras are coherently equipped with the structure of complete Elgot algebras for the parametrized monads obtained from generalized coalgebraic resumption monads.

Keywords: Complete Elgot monad, complete Elgot algebra, resumption monad, uniform iteration

1 Introduction

One traditional use of monads in computer science, stemming from the seminal thesis of Lawvere [20], is as a tool for algebraic semantics where monads arise as a high-level metaphor for (clones of) equational theories. More recently, Moggi proposed to associate monads with *computational effects* and use them as a generic tool for denotational semantics [22], which later had a considerable impact on the design of functional programming languages, most prominently Haskell [1]. Finally, in the first decade of the new millennium, Plotkin and Power reestablished the

† Full version is available at http://arxiv.org/abs/1603.02148

⋆ Supported by Deutsche Forschungsgemeinschaft (DFG) under project SCHR 1118/8-1

1 Email: Sergey.Goncharov@fau.de

2 Email: mail@stefan-milius.eu Supported by Deutsche Forschungsgemeinschaft (DFG) under project MI 717/5-1

3 Email: Christoph.Rauch@fau.de

http://dx.doi.org/10.1016/j.entcs.2016.09.036

connection between computational monads and algebraic theories in their theory of *algebraic effects* [23,24].

We use the outlined view to study the notion of *iteration* as a concept that has a well-established algebraic meaning and which is very relevant in the context of computational effects. On the technical level our present work can be viewed as a continuation of the previous extensive work on monads with iteration [2,5,7] having its roots in the work of Elgot [12] and Bloom and Ésik [10] on iteration theories.

More specifically, we are concerned with a particular construction on monads: given a monad $\mathbb{T}$ and a functor Σ, we assume the existence of the coalgebra

$$T_\Sigma X = \nu\gamma.\, T(X + \Sigma\gamma) \qquad (\bigstar)$$

for each object X (these final coalgebras exist under mild assumptions on T, Σ, and the base category). It is known [28] that T_Σ extends to a monad $\mathbb{T}_\Sigma$ and we call the latter the *generalized coalgebraic resumption monad*.

Intuitively, ($\bigstar$) is a generic semantic domain for systems combining *extensional* (via $\mathbb{T}$) and *intensional* (via Σ) features with iteration. To make this intuition more precise, consider the following simplistic

Example 1.1 Let $A = \{a, b\}$ be an alphabet of *actions*. Then the following system of equations specifies *processes* x_1, x_2, x_3 of *basic process algebra (BPA)*:

$$x_1 = a \cdot (x_2 + x_3) \qquad x_2 = a \cdot x_1 + b \cdot x_3 \qquad x_3 = a \cdot x_1 + \checkmark$$

We can think of this specification as a map $P \to T(\{\checkmark\} + \Sigma P)$ where $P = \{x_1, x_2, x_3\}$, $\Sigma = A \times -$ and $T = \mathcal{P}_\omega$ is the finite powerset monad. Using the standard approach [26] we can *solve* this specification by finding a map $P \to T_\Sigma\{\checkmark\}$ that assigns to every x_i the corresponding semantics over the domain of possibly non-wellfounded trees $T_\Sigma\{\checkmark\} = \nu\gamma.\mathcal{P}_\omega(\{\checkmark\} + A \times \gamma)$. The crucial fact here is that the original system is *guarded*, i.e. every recursive call of a variable x_i is preceded by an action. This implies that the given recursive system has a unique solution.

If the guardedness assumption is dropped, solutions may fail to be unique, but it is possible to introduce a notion of *canonical solution* if the Kleisli category of the monad $\mathbb{T}$ is enriched in the category of complete partial orders, or more generally, if $\mathbb{T}$ is a *complete Elgot monad*. A monad $\mathbb{T}$ is a complete Elgot monad if it is equipped with an iteration operator that assigns to every morphism of the form $f : X \to T(Y + X)$ a *solution* $f^\dagger : X \to TY$ satisfying a certain well-established set of equational axioms of iteration and also *uniformity* [27] (e.g. $\mathcal{P}_\omega$ is not a complete Elgot monad, but the countable powerset monad $\mathcal{P}_{\omega_1}$ is). The central result of the recent work [14] is that whenever $\mathbb{T}$ is a complete Elgot monad then so is the transformed monad ($\bigstar$). In particular, this allows for canonical solutions of recursive equations over processes (in the sense of Example 1.1) whenever recursive equations over $\mathbb{T}$ are solvable.

In the present paper we investigate the relationship between guarded and unguarded iteration, which are implemented via generalized coalgebraic resumption