From Equations to Distinctions: Two Interpretations of Effectful Computations, an extended list of examples

Niels Voorneveld*
Tallinn University of Technology
niels.voorneveld@taltech.ee

September 2020

This living document is an informal addendum to the MSFP 2020 paper "From Equations to Distinctions: Two Interpretations of Effectful Computations" [2]. This paper concerns itself with behavioural descriptions of algebraic effects. In particular, it shows how we can use axiomatic equations an inequations to construct an Eilenberg Moore algebra on the cofree monad over the effect operations. This algebra is then suitable for specifying a congruent notion of program equivalence for functional languages with effects, such as Plotkin's call-by-value PCF and Paul Levy's call-by-push-value.

In this document, we will look for various examples of algebraic effects, which algebra will be constructed from particular sets of axioms. We will use $\mathbb{A}$ to denote the carrier of this algebra, denoting the set of truth values. These are given by the quotient of continuation free trees, which have no leaves, but do have two nodes of zero arity, denoting failure/falsehood/divergence and success/truth/termination by $\bot$ and $\top$ respectively. The algebra will be given by α, where for each example we will show how it behaves locally over the effect operations. Note that we include infinite trees, and the equational theory for those is mainly based on admissibility (approximations). In particular, an infinite tree without any leaves or success nodes is automatically equal to $\bot$.

Unless stated otherwise, the carrier sets derived below are complete lattices and the algebras are ω-continuous. As such, according to previous work [1], they are suitable for constructing a congruent notion of program equivalence, both in the form of a Quantitative logic and in the form of Applicative similarity/bisimilarity via the definition of a relator.

This document only contains conclusions, not derivations. So it should only be used as an overview of examples. Some interesting derived equations are mentioned in some cases. Keep in my that this is not a complete list of examples, just a selection. I may add more examples in due time. Please send me an e-mail

*This research was supported by the ESF funded Estonian IT Academy research measure (project 2014-2020.4.05.19-0001).

if you are interested in whether your example of algebraic effect (combination) fits in this framework, or if you want more info about an example (e.g. what the induced relator looks like).

No operations

- *Pure computation.*

  No equations.

  $\mathbb{A} := \{\bot \leq \top\}$, algebra is the obvious one.

- *Unity*, for any set of operations.

  Assume $\top \leq \bot$, then $\mathbb{A} = \mathbf{1}$.

Continuation-free operations, for any $e \in E$ an operation $\mathsf{expt}_e : 0$.

- *Discrete exceptions/errors*

  No equations.

  $\mathbb{A} = E + \{\bot, \top\}$ where $\forall e \in E.\ \bot \leq \mathsf{expt}_e \leq \top$.

- *Ordered exceptions/errors*, assume E has an order.

  For any $e_0, e_1 \in E$ such that $e_0 \leq e_1$, assume $\mathsf{expt}_{e_0} \leq \mathsf{expt}_{e_1}$.

Unary operation examples, $\mathsf{tick} : 1$.

- *Bleep machine, or unit printer*

  No equations.

  $\mathbb{A} := \mathbb{N} + \mathbb{N}$ where $\mathit{inl}(n) \leq \mathit{inl}(n+1) \leq \mathit{inr}(m+1) \leq \mathit{inr}(m)$.

  $\alpha(\mathsf{tick}(x)) = \alpha(x) + 1$, whatever side you are on.

- *Cost*

  Axiom: $\mathsf{tick}(x) \leq x$

  Derivable: $\bot = \mathsf{tick}(\bot)$.

  $\mathbb{A} = \{\infty\} + \mathbb{N}$ where $\infty \leq \mathit{inr}(n+1) \leq \mathit{inr}(n)$.

  $\alpha(\mathsf{tick}(x)) = \alpha(x) + 1$ where $\infty + 1 = \infty$.

- *Lights on*

  Axiom: $\mathsf{tick}(\mathsf{tick}(x)) = \mathsf{tick}(x)$.

  $\mathbb{A} = \{\bot \leq \mathsf{tick}(\bot) \leq \mathsf{tick}(\top) \leq \top\}$.

- *A non-example, an unsteady equation*

  Axiom: $\mathsf{tick}(\mathsf{tick}(\mathsf{tick}(x))) = \mathsf{tick}(x)$.

  Derivable: $\mathsf{tick}(\bot) = \mathsf{tick}(\mathsf{tick}(\bot))$ and $\mathsf{tick}(\top) = \mathsf{tick}(\mathsf{tick}(\top))$.

  However, $\mathsf{tick}(x) = \mathsf{tick}(\mathsf{tick}(x))$ is not derivable! But it should be if the induced algebraic relation is to be base-valued...

Binary choice examples, $\mathsf{or} : 2$.

- *Binary input, or bitstream reader*

  No equations.

  $\mathbb{A} = \{f : 2^* \to \{\bot \le \circ \le \top\} \mid \forall \sigma \in 2^*.\ f(\sigma) \neq \circ \Rightarrow f(\sigma 0) = f(\sigma) = f(\sigma 1)\}$

  $\alpha(\mathsf{or}(x,y))([]) = \circ$, $\alpha(\mathsf{or}(x,y))(0\sigma) = \alpha(x)(\sigma)$, $\alpha(\mathsf{or}(x,y))(1\sigma) = \alpha(y)(\sigma)$.[1]

- *Static state*

  Axioms: $\mathsf{or}(x,x) = x$, $\mathsf{or}(\mathsf{or}(x,y),z) = \mathsf{or}(x,z) = \mathsf{or}(x,\mathsf{or}(y,z))$.

  $\mathbb{A} = \{L,R\} \to \mathbf{B}$, $\alpha(\mathsf{or}(x,y))(L) = \alpha(x)(L)$, $\alpha(\mathsf{or}(x,y))(R) = \alpha(y)(R)$.

- *Nondeterminism (neutral or convex)*

  Axioms: $\mathsf{or}(x,x) = x$, $\mathsf{or}(x,y) = \mathsf{or}(y,x)$, $\mathsf{or}(\mathsf{or}(x,y),z) = \mathsf{or}(x,\mathsf{or}(y,z))$.

  Derivable: $\mathsf{or}(x,\mathsf{or}(x,\mathsf{or}(x,\dots))) = \mathsf{or}(x,\bot)$ (by admissibility).

  $\mathbb{A} = \{\bot \le \diamond \le \top\}$, $\alpha(\mathsf{or}(x,y)) = \alpha(x)$ if $\alpha(x) = \alpha(y)$, otherwise its $\diamond$.

\+ *Angelic nondeterminism*

  We add the axiom $x \le \mathsf{or}(x,y)$. This unifies $\diamond$ and $\top$.

\+ *Demonic nondeterminism*

  Instead, we add the axiom $\mathsf{or}(x,y) \le x$. This unifies $\diamond$ and $\bot$.

- *Probability*

  Axioms: $\mathsf{or}(x,x) = x$, $\mathsf{or}(x,y) = \mathsf{or}(y,x)$, $\mathsf{or}(x,\mathsf{or}(x,\mathsf{or}(x,\dots))) = x$, $\mathsf{or}(\mathsf{or}(x,y),\mathsf{or}(z,w)) = \mathsf{or}(\mathsf{or}(x,z),\mathsf{or}(y,w))$.

  $\mathbb{A} = [0,1]$, $\alpha(\mathsf{or}(x,y)) = (\alpha(x) + \alpha(y))/2$.

- *Bit-toggle, the minimalist global store*

  Axioms: $\mathsf{or}(\mathsf{or}(z,x),y) = \mathsf{or}(\mathsf{or}(w,x),y)$, $\mathsf{or}(x,\mathsf{or}(y,z)) = \mathsf{or}(x,\mathsf{or}(y,w))$, $\mathsf{or}(\mathsf{or}(x,y),\mathsf{or}(y,z)) = y$.

  Derivable: $\mathsf{or}(\mathsf{or}(x,\mathsf{or}(y,z)),w) = \mathsf{or}(y,w)$.

  $\mathbb{A} = \{0,1\} \to \mathbf{B}$, $\alpha(\mathsf{or}(x,y))(0) = \alpha(x)(1)$, $\alpha(\mathsf{or}(x,y))(1) = \alpha(y)(1)$.

- *List, or the broke-pay operation*

  Axiom: $\mathsf{or}(\mathsf{or}(x,y),z) = \mathsf{or}(x,z)$.

  $\mathbb{A} = \{\bot,\top\}^*_{\neq []}$, where $\sigma\bot \le \sigma\tau\bot$, $\sigma\bot\tau \le \sigma\top\tau$, $\sigma\tau\top \le \sigma\top$.

  $\alpha(\mathsf{or}(x,y)) = \mathsf{head}(\alpha(x))\alpha(y)$.

[1] For a set X, we denote X^* for the set of finite lists over X.

More sophisticated versions of the above effects

- *Traditional binary global store*

  Operations: $\mathsf{lookup} : 2$, $\mathsf{update}_0 : 1$, $\mathsf{update}_1 : 1$.

  Axioms: $\mathsf{lookup}(x, x) = x$, $\mathsf{lookup}(\mathsf{update}_0(x), \mathsf{update}_1(y)) = \mathsf{lookup}(x, y)$, $\mathsf{update}_0(\mathsf{lookup}(x, y)) = \mathsf{update}_0(x)$, $\mathsf{update}_1(\mathsf{lookup}(x, y)) = \mathsf{update}_1(y)$.

  Derivable: $\mathsf{lookup}(\mathsf{lookup}(x, y), z) = \mathsf{lookup}(x, z) = \mathsf{lookup}(x, \mathsf{lookup}(y, z))$, $\mathsf{update}_i(\bot) = \bot$, $\mathsf{update}_i(\top) = \top$.

  $\mathbb{A} = \{0, 1\} \to \mathbf{B}$,

  $\alpha(\mathsf{lookup}(x_0, x_1))(i) = \alpha(x_i)(i)$, $\alpha(\mathsf{update}_i(x))(j) = \alpha(x)(i)$.

- *Print*

  Operations: A set of printing messages M, $\forall m \in M$. $\mathsf{print}_m : 1$.

  No equations.

  $\mathbb{A} = M^* \times \{\bot, \top\}$, $(\sigma, \bot) \leq (\sigma m, \bot)$, $(\sigma, \bot) \leq (\sigma, \top)$, $(\sigma m, \top) \leq (\sigma, \top)$.

  $\alpha(\mathsf{print}_m(x)) = (m\ \mathsf{first}(\alpha(x)), \mathsf{second}(\alpha(x)))$.

- *Printing without linebreaks*

  Operations: A set of characters C, $\forall \sigma \in C^*$. $\mathsf{print}_\sigma : 1$.

  Axioms: $\forall \sigma \in C^*. \forall c \in C$. $\mathsf{print}_\sigma(\mathsf{print}_c(x)) = \mathsf{print}_{\sigma c}(x)$.

  $\mathbb{A} = C^* \times \{\bot, \top\}$ with same order and similar algebra as previous example.

- *Input/Output*

  Operations: A set of inputs I and outputs O, $\mathsf{read} : I$, $\forall o \in O.\mathsf{write}_o : 1$.

  No equations, so $\mathbb{A} = T\emptyset$. There are ways to better represent this carrier set, considering certain subsets of traces. Very similar to tactics in game theory.

- *Versus nondeterminism*

  Two operations: $\mathsf{angor} : 2$, demor.

  Axioms of angelic nondeterminism for angor, and demonic nondeterminism for demor, plus $\mathsf{angor}(\mathsf{demor}(x, y), z) = \mathsf{demor}(\mathsf{angor}(x, z), \mathsf{angor}(y, z))$, and $\mathsf{demor}(\mathsf{angor}(x, y), z) = \mathsf{angor}(\mathsf{demor}(x, z), \mathsf{demor}(y, z))$.

  $\mathbb{A} = \mathbf{B}$, $\alpha(\mathsf{angor}(x, y)) = \alpha(x) \vee \alpha(y)$, $\alpha(\mathsf{demor}(x, y)) = \alpha(x) \wedge \alpha(y)$.

- *Jump, or exception catching*

  Operations: Set of exceptions E, $\forall e \in E$. $\mathsf{catch}_e : 2$, $\mathsf{expt}_e : 0$.

  Axioms: $\mathsf{catch}_e(\mathsf{expt}_e, x) = x$, $\mathsf{catch}_e(\mathsf{expt}_{e'}, x) = \mathsf{expt}_{e'}$ if $e \neq e'$, $\mathsf{catch}_e(x, x) = x$, $\mathsf{catch}_e(x, \mathsf{expt}_e) = x$, $\mathsf{catch}_e(\bot, x) = \bot$, $\mathsf{catch}_e(\top, x) = \top$.

  $\mathbb{A} = E + \{\bot, \top\}$ where $\forall e \in E$. $\bot \leq \mathsf{expt}_e \leq \top$.

  $\alpha(\mathsf{expt}_e) = e$, $\alpha(\mathsf{catch}_e(x, y)) = \alpha(y)$ if $\alpha(x) = e$, $\alpha(\mathsf{catch}_e(x, y)) = \alpha(x)$ if $\alpha \neq x$.

Combinations of effects

- *Probability with cost*

  Operations: $\mathsf{or} : 2$, $\mathsf{tick} : 1$.

  Axioms: Probability + Cost + $\mathsf{tick}(\mathsf{or}(x, y)) = \mathsf{or}(\mathsf{tick}(x), \mathsf{tick}(y))$.

  $\mathbb{A} = \{f : \mathbb{N} \to [0, 1] \mid f \quad \text{increasing}\}$, $\alpha(\mathsf{or}(x, y))(n) = (\alpha(x)(n) + \alpha(y)(n))/2$, $\alpha(\mathsf{tick}(x))(0) = 0$, $\alpha(\mathsf{tick}(x))(n + 1) = \alpha(x)(n)$.

- *Probability with global store*

  Operations: $\mathsf{or} : 2$, $\mathsf{lookup} : 2$, $\mathsf{update}_0 : 1$, $\mathsf{update}_1 : 1$.

  Axioms: Probability + Global store + distribution laws.

  $\mathbb{A} = \{0, 1\} \to [0, 1]$,

  $\alpha(\mathsf{or}(x, y))(i) = (\alpha(x)(i) + \alpha(y)(i))/2$, $\alpha(\mathsf{lookup}(x_0, x_1))(i) = \alpha(x_i)(i)$, $\alpha(\mathsf{update}_j(x))(i) = \alpha(x)(j)$.

- *Probability with nondeterminism*

  Operations: $\mathsf{prob} : 2$, $\mathsf{nond} : 2$.

  Axioms: Probability for prob + nondeterminism for nond
  + $\mathsf{prob}(\mathsf{nond}(x, y), z) = \mathsf{nond}(\mathsf{prob}(x, z), \mathsf{prob}(y, z))$.

  $\mathbb{A} = \{(p, q) \in [0, 1] \mid p \geq q\}$,

  $(p, q) = \alpha(x)$ and $(p', q') = \alpha(y)$, then $\alpha(\mathsf{prob}(x, y)) = ((p + p')/2, (q + q')/2)$ and $\alpha(\mathsf{nond}(x, y)) = (\max(p, p'), \min(q, q'))$.

Note: The last example is not unit-valued. However, the other examples do seem to be. As discussed in the paper, being unit-valued makes it so the quantitative algebra can be decomposed into Boolean algebras while preserving the induced algebraic relation.

References

[1] Niels Voorneveld. Quantitative logics for equivalence of effectful programs. volume 347, pages 281 – 301, 2019. Proceedings of the Thirty-Fifth Conference on the Mathematical Foundations of Programming Semantics.

[2] Niels Voorneveld. From equations to distinctions: Two interpretations of effectful computations. *Electronic Proceedings in Theoretical Computer Science*, 317:1–17, May 2020.