Available online at [www.sciencedirect.com](http://www.sciencedirect.com/)

[Electronic Notes in Theoretical Computer Science 341 (2018) 277–295](https://doi.org/10.1016/j.entcs.2018.11.014)

[www.elsevier.com/locate/entcs](http://www.elsevier.com/locate/entcs)

Near Distributive Laws

Ernie Manes[1](#_bookmark0)

*Department of Mathematics and Statistics University of Massachusetts at Amherst USA*

Phil Mulry[2](#_bookmark0)

*Department of Computer Science Colgate University Hamilton NY, USA*

**Abstract**

Monads and their compositions can sometimes be generated from simpler data types and without neces- sarily requiring any monad axioms. Free monads and monad approximations provide two approaches to overcoming the constraints required by monad composition laws while generating near distributive laws.

*Keywords:* monad composition, free monad, monad approximation, near distributive law.

# Introduction

This paper continues our study of monad composition in [[10](#_bookmark31)], [[11](#_bookmark32)]. We will use the same notations as in the second of these papers. We work in a category *V*.

In working with monads in a programming language, there are two problems: It may be hard to define a monad using the data types available to the programmer, and it may be difficult to verify the monad axioms. The second of these is a very common situation in monad composition.

Given monads (*H, μ, η*) and *K, ν, ρ*) their composition should have form (*KH, τ, ρη*). The problem is that there is no obvious *τ* . The solution is to pro- vide a natural transformation *λ* : *HK → KH* which allows *τ* to be defined as

KλH νμ

*τ* = *KHKH −−→ KKHH −−→ KH*

1 Email: [egmanes@gmail.com](mailto:myuserid@mydept.myinst.myedu)

2 Email: [pmulry@colgate.edu](mailto:couserid@codept.coinst.coedu)

<https://doi.org/10.1016/j.entcs.2018.11.014>

1571-0661/© 2018 The Author(s). Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

The axioms on *λ* equivalent to rendering (*KH, τ, ρη*) a monad were discovered by [[3](#_bookmark26)] and are as follows.

*H Hρ* ) *HK*

( *Hν HKK*

(*DL A*)

*λ*

(*DL B*)

v*λK KHK*

*ρH*

v

v*Kλ*

*KH* ( *KKH*

*νH*

*K ηK* ) *HK*

( *μK*

*HHK*

(*DL C*)

*λ*

(*DL D*)

v*Hλ HKH*

*Kη*

v

v*λH*

*KH* ( *KHH*

*Kμ*

Axioms (DL C, DL D) hold if and only if *K* lifts through the category *V***H** of Eilenberg-Moore algebras of *H* and we say *λ* is a **near distributive law** in this case.

In practice, such axioms present an obstruction to the programmer. We recall [[4](#_bookmark27), Page 34]:

“all polymorphic functions in functional programming are natural transforma- tions”.

Thus the axioms are the obstruction and not the requirement of naturality. In this paper we present two approaches to getting around the obstruction and, in the process, discover various near distributive laws. In the first case, we consider endofunctors *H, K* and arbitrary natural transformations *λ* : *HK → KH* where there are no axioms on *λ*. If *H* generates a free monad **H**@ then *λ* induces a near distributive law *λ*@ : *H*@*K → KH*@ and say *λ generates λ*@. We consider in detail near distributive laws for a common class of free monads namely those generated by algebraic signatures.

A different approach involves defining the notion of a **pre-monad** in *V* to be (*H, η, μ*) with *H* : *V → V* an endofunctor and *μ* : *HH → H*, *η* : id *→ H* natural transformations, with no further axioms. Pre-monads retain a surprising amount of structure. Analogous to the property of generating a free monad, we shall see that a pre-monad **H** usually has a monad approximation **H**. We show that if (*K, m, e*) isa pre-monad, then a near distributive law *λ* : *HK → KH* induces a near distributive law *λ*^ : *H*^ *K → KH*^ .

^

# Free Monads

Our most relaxed model of a monad is a functor *H* : *V → V*. In standard situations the **free monad generated by** *H*, (*H*@*, μ, η*; *ι*) exists where **H**@ = (*H*@*, μ, η*) isa monad in *V* and *ι* : *H → H*@ is a natural transformation, subject to the universal property [[2]](#_bookmark25)

*H*  *ι* ) *H*@

(*H*@*, μ, η*)

*α* *ψ*

v

*K*

*ψ*

v

(*K, ν, ρ*)

that if (*K, ν, ρ*) is a monad in *V* and *α* : *H → K* is a natural transformation then there exists a unique monad map *ψ* as shown with *ψι* = *α*.

**Example 2.1** Assume that *V* has finite powers. For a finite ordinal *i ≥* 1, let *H*i : *V → V* be the functor *H*i*X* = *X*i, the usual *i*-ary power functor. When *V* = **Set**, the data type *H*@*X* is the set of all trees in which every node is either an element of *X*, denoted *L*i*x* (if it is a leaf) or has *i* subtrees beneath it, denoted *B*i*t*1 *··· t*i *∈ H*@*X*. The natural transformation *η*X : *X → H*@*X* maps *x* to *L*i*x*

i

i i

while *μ*X : *H*@*H*@*X → H*@ maps *L*i*t* to *t* and *B*i *t*1 *... t*i to *B*i (*μ*X *t*1) *...* (*μ*X *t*i).

i i i

In what follows, we consider only *H* for which **H**@ exists.

**Definition 2.2** An *H***-algebra** is a pair (*X, δ*) where *δ* : *HX → X* in *V*. An

*H***-homomorphism** *f* : (*X, δ*) *→* (*Y, ϵ*) of *H*-algebras must satisfy

*HX Hf* ) *HY*

*δ ϵ*

v v

*X* ) *Y*

*f*

It is evident that idX : (*X, δ*) *→* (*X, δ*) is an *H*-homomorphism and that *H*- homomorphisms are closed under composition. This gives rise to a category *V*H of *H*-algebras with underlying functor *V*H *→ V*.

**Theorem 2.3** *[*[*2*](#_bookmark25)*] If V*H *→V is monadic, V*H *is isomorphic over V to the category of Eilenberg-Moore algebras V***H**@ *, where* **H**@ *is the free monad generated by H. The isomorphism* Φ: *V***H**@ *→ V*H *is given by*

@ ξ ι*X* @ ξ

Φ(*X, H*

*X −−→ X*) = (*X, HX −−→ H X −−→ X*)

# Functorial Lifts

**Definition 3.1** Let **H** = (*H, μ, η*) bea monad in *V*, and let *K* : *V → V* be a functor. A functor *K*٨ : *V***H** *→ V***H** is a **functorial lift** of *K* through the Eilenberg-Moore category *V***H** if the following square commutes:

*K*٨)

**H**

**H**

*V* *V*

v v

*V* ) *V*

*K*

The following result is due to [[1](#_bookmark24)]. Also, see [[7](#_bookmark29)].

**Theorem 3.2** *For* **H** = (*H, μ, η*) *a monad in V and K* : *V → V a functor, functo- rial lifts K*٨ : *V***H** *→ V***H** *are in bijective correspondence with natural transformations λ* : *HK → KH which satisfy (DL C) and (DL D). The correspondences are*

٨ λ*X* Kξ

1. *K*

*and*

(*X, ξ*)= (*KX, HKX −−−−→ KHX −−−−→ KX*)

HKη*X* ω*X*

1. *λ*X = *HKX −−−−→ HKHX −−−−→ KHX*

*X*

ω

*where K*

٨(*HX, μ*X)= (*KHX, HKHX −−→ KHX*)*. Further,*

λ*HX*

Kμ*X*

1. *ω*X = *HKHX −−−−→ KHHX −−−−→ KHX*

It is immediate that *λ*X : (*HKX, μ*KX) *→ K*٨(*HX, μ*X) isa **H**-homomorphism; the homomorphism diagram is precisely (DL D). It follows (using (DL C)) that *λ*X is the unique homomorphic extension of *Kη*. There is more than one possi- ble *λ*, however, because there is more than one possible **K**-algebra structure for *K*٨(*HX, μ*X).

**Definition 3.3** Let *H* : *V → V* generate a free monad **H**@ and let *K* : *V → V* be a functor. Let *K*٨ : *V***H**@ *→ V***H**@ be a functorial lift of *K* with classifying natural transformation *λ*@ : *H*@*K → KH*@ as in Theorem [3.2](#_bookmark3). We say *K*٨ is a **flat** functorial lift if there exists a natural transformation *λ* : *HK → KH* such that the following square commutes.

*HK*  *ιK* ) *H*@*K*

*λ λ*@ ([3*.*3](#_bookmark4))

v v

)

*KH Kι KH*@

We then say that *λ* **generates** *K*٨, or *λ* **generates** *λ*@, and when *K* is a monad that *λ*@ is a **flat near distributive law**.

**Theorem 3.4** *Given H, K* : *V → V such that* **H**@ *exists, every natural transfor- mation λ* : *HK → KH generates a flat functorial lift of K through* **H**@*.*

**Proof.** Given *λ*, define *K†* : *V*H *→ V*H over *V* by

*†* λ*X* Kδ

*K* (*X, δ*) = (*KX, HKX −−−−→ KHX −−−−→ KX*)

If *f* : (*X, δ*) *→* (*Y, ϵ*) is an *H*-homomorphism, the diagram

*HKX*

*λ*X ) *KHX*  *Kδ*) *KX*

*HKf*

v

*KHf Kf*

v v

) )

*HKY λ*Y

*KHY*

*Kϵ KY*

shows that *Kf* : *K†*(*X, δ*) *→ K†*(*Y, ϵ*) is again an *H*-homomorphism. In the nota- tions of Theorem [2.3](#_bookmark2) we then have the functorial lift

*−−−−→V*

*−−−−→V*

*K*٨ = *V***H**@ Φ

*−−−−→V*

H K*†*

H Φ*−*1 **H**@

We leave the remaining details to the reader. *2*

**Corollary 3.5** *Given H, K* : *V → V where* **K** *is a monad and* **H**@ *exists, then every natural transformation λ* : *HK → KH generates a flat near distributive law λ*@ : *H*@*K → KH*@*.*

# Near Distributive Laws for Free Monads

* 1. *Near Distributive Laws via Generic Prestrengths*

The notion of prestrength on an endofunctor *F* of a category was defined and used in [[11](#_bookmark32)] and [[12](#_bookmark33)] as part of the process of working with Kleisli strength. For fixed *n ≥* 1, a prestrength of order *n* on the functor *F* is a natural transformation Γn : *FX*1 *×· · · × FX*n *→ F* (*X*1 *×· · · × X*n). We note immediately that a special case of a prestrength of order *n* on the functor *F* is a natural transformation Γn : *H*n*F → F H*n, where *H*n is the *n*-ary power functor of Example [2.1](#_bookmark1). We exploit the existence of what could be suitably called a monad-induced *generic prestrength* to derive classes of near distributive laws on free monads **H**@. Later, in Section [4.3](#_bookmark11), we will identify alternative non-generic kinds of prestrengths which generate in turn different near distributive laws.

n

**Lemma 4.1** *For any monad* **K** = (*K, ν, ρ*) *in* **Set** *there exists a generic prestrength*

Γn : *KX*1 *×* *KX*n *→ K*(*X*1 *×* n) *of order n ≥* 1*.*

**Proof.** Let **K** = (*K, ν, ρ*) be a monad in **Set**. We show for any given *n ≥* 1 there exists a natural transformation Γn : *KX*1 *× ...KX*n *→ K*(*X*1 *× ...X*n). Suppose there exists a natural transformation in two variables

Γ*XYKX × KY − −→ K*(*X × Y* )

*−−*

Then for *n* = 1 define Γ1 = idX1 , and for *n* = 2 let Γ2 = Γ*X X*

. Proceeding

1 2

inductively, if Γi is natural, we obtain a natural transformation Γi+1 by

id*X*1 *×*Γ*i*

Γ*X*1(*X*2 *... ×Xi*+1)

*KX*1 *×* (*KX*2 *... × KX*i+1) *−−−−−−→ KX*1 *× K*(*X*2 *... × X*i+1) *−−−−−−−−−−→ K*(*X*1 *× X*2 *... × X*i+1)

To construct Γ2, for *x ∈ X* let inx : *Y → X × Y* be defined by inx(*y*)= (*x, y*). It is

obvious that the following square commutes for each *f* : *X → X*1, *g* : *Y → Y*1:

*Y* inx

*g*

) *X × Y*

*f × g*

v )

*Y*1 infx

v

*X*1 *× Y*1

Define *δXY* : *X × KY → K*(*X × Y* ) by *δXY* (*x, τ* )= (*K* inx) *τ* . From the preceding square and the functoriality of *K* we obtain

*X × KY*

*δXY*

) *K*(*X × Y* )

*f × Kg*

v

)

*K*(*f × g*)

v

*X*1 *× KY*1

*δX*1*Y*1

*K*(*X*1 *× Y*1)

At this stage, we need that **K** is a monad. Any function from *X* to a **K**-algebra admits a unique **K**-homomorphic extension *f* # from the free algebra (*KX, μ*X).

Define the desired Γ*XY*

: *KX × KY → K*(*X × Y* ) by Γ*XY*

(*σ, τ* )= *δXY*

(*·,τ* )#*σ*. The

desired naturality square amounts to the commutativity of

(*δ* (*·,τ* ))#

*XY*

*KX*

) *K*(*X × Y* )

*Kf K*(*f × g*)

v v

)

*KX*1

(*δX*1*Y*1

(*·,* (*Kf* )*τ* ))#

*K*(*X*1 *× Y*1)

for each *τ ∈ KY* . Since each of the four maps in this square isa **K**-homomorphism, it suffices to check commutativity restricted to the generators *ρ*X and this is clear from the square for *δXY* immediately above. *2*

* 1. *Amenable Monads*

We know of no nontrivial monad which admits a distributive law with every monad. This places some constraint on the use of distributive laws in programming. We consider instead monads which admit near distributive laws with every monad, calling these *amenable* and provide examples.

**Definition 4.2** A monad **H** in *V* is **amenable** if for every monad **K** in *V*, *K* has a functorial lift through *V***H**.

**Proposition 4.3** *The monads* **H**@ *in* **Set** *of Example* [*2.1*](#_bookmark1) *are amenable.*

**i**

**Proof.** Let **K** = (*K, ν, ρ*) bea monad in **Set**. By Lemma [4.1](#_bookmark6) there exists a generic natural transformation Γi : *H*i*K → KH*i for every *i ≥* 1 and so by Corollary [3.5](#_bookmark5) we are done. *2*

**Example 4.4** Let (*C, e, ∗*) bea monoid and let **K** be the reader monad *KX* = *C ×*

*X*. Then *λ* = Γ2 : *H*2*K → KH*2 in Proposition [4.3](#_bookmark7) becomes Γ2((*c*1*, x*1)*,* (*c*2*, x*2)) = (*c*1 *∗ c*2*,* (*x*1*, x*2)). Acting on a binary tree *t* of type *H*@*KX*, *λ*@(*t*)= (*p, t∗*) where *p* is the product of the *c*is found in the leaves and *t∗* is the corresponding tree in *H*@*X* consisting only of the elements of *X*.

2

2

**Example 4.5** When *K* is itself a free monad of the form *H*@, we can give a recursive

j

construction of the functorial lift of *K* = *H*@

j

@

through **Set** *i*

H

defining the near

distributive law *λ* : *H*@*H*@ *→ H*@*H*@ in cases. Details are straightforward and left

i j j i

to the reader (see Example [2.1](#_bookmark1) for notation). For *i, j ≥* 1:

*λ*(*L*i*L*j*a*)= *L*j*L*i*a*

*λL*i(*B*j*t*1 *··· t*j)= *B*j(*λL*i*t*1) *···* (*λL*i*t*j)

*λB*i(*t*1 *··· t*i)= (*H*@*B*i)Γi((*λt*1)*, ··· ,* (*λt*i))

j

**Proposition 4.6** *Let V have small coproducts, let* (*H*α) *be a small family of endo- fuctors and let H* = *H*α *be the pointwise coproduct. Assume that the free monads* **H**@*,* **H**@ *exist. Then if each* **H**@ *is amenable, so is* **H**@*.*

α α

**Proof.** An **H**-algebra is determined by a family (*δ*α : *H*α*X → X*). Let **K** be a

δ*α* є*α*

monad in *V* and let (*X, H*α*X −−→ X*) *'→* (*KX, H*α*KX −−→ X*) under a functorial

lift of *K* through *V***H***α* . The remaining details are clear. *2*

**Example 4.7** Let Σ be a finitary operator domain, that is, a disjoint sequence (Σn) of (possibly empty) sets. A Σ-algebra (as conventionally defined in universal algebra) is (*X, δ*) where *X* is a set and *δ* = (*δ*σ : *σ ∈* Σ) with *δ*σ : *X*n *→ X* if *σ ∈* Σn. Consider the coproduct functor

*H*Σ*X* = *X*n

σ*∈*Σ*n*

so that an *H*Σ-algebra is the same thing as a Σ-algebra.

A variety of universal algebras is obtained from **Set**HΣ by imposing equations.

*H*@*X* is the usual free Σ-algebra generated by *X*. It is immediate from Propositions

Σ

[4.3](#_bookmark7) and [4.6](#_bookmark10) that **H**@ is an amenable monad in **Set**.

Σ

* 1. *Prestrengths and Flat near distributive Laws*

In this section we consider a different class of flat near distributive laws which generally differ from those of the previous section.

**Lemma 4.8** *For any monad* **K** = (*K, ν, ρ*) *in* **Set***, if there exists a natural trans- formation γ* : *K → id then there exists a prestrength* Γi : *KA*1 *× ... × KA*i *→ K*(*A*1 *× ... × A*i) *for any i ≥* 1*.*

**Proof.** The construction is simple: for *i* = 1 define Γ1 = *ρ ◦ γ*. If *i ≥* 2 then Γi = *ρ◦*(*γ×.. .×γ*). Since in each case Γi is a composition of natural transformations we are done. *2*

**Proposition 4.9** *For any monad* **K** = (*K, ν, ρ*) *with natural transformation γ* :

*K → id, there exists a flat near distributive law λ*@ : **H**@*K → K***H**@*.*

**i** **i**

**Proof.** For any *i ≥* 1, the prestrength Γi of the previous lemma generates a natural transformation *H*i*K → KH*i and so the result follows immediately from Corollary

[3.5](#_bookmark5). *2*

**Example 4.10** For *j ≥* 1 let *π*j denote the *j*-th projection natural transformation *H*j *→* id. By the previous proposition this generates a flat near distributive law *λ*@ : **H**@*H*@ *→ H*@**H**@ which generally differs from that of Example [4.5](#_bookmark9).

**i** j j **i**

**Example 4.11** Let **K** be the reader monad of Example [4.4](#_bookmark8) with monoid (*C, e, ∗*). Define *γ* : *K →* id by *γ*(*c, a*)= *a*. Such *γ* is natural, generating Γn : *KA*1*×...KA*n *→ K*(*A*1 *× ...A*n). For the monad of non-empty lists **L**, the resulting flat distributive law *λ*@ : **L**(*C × A*) *→ C ×* **L***A* takes [(*c*1*, a*1)*, ...*(*c*n*, a*n)] to (*e,* [*a*1*, ...a*n]).

* 1. *Uniformly branching trees and non-flat near distributive laws.*

For the free monad **H**@, *H*@*X* consists of trees in which every non-leaf has *i*

**i** i

branches. Due to their particular structure, these trees also generate a class of

**j**

(*not necessarily flat*) near distributive laws of **H**@

**i**

over **H**@

for which very little

underlying data is either created or destroyed in the process. Significantly these near distributive laws do not arise via flat liftings, but rather arise directly from the monad structure on *H*@.

i

Recall that an algebra for *H*@ is generated by (*A,* [ ]i), where [ ]i : *A*i *→ A* is an *i*-ary operation on *A*. For *i, j ≥* 1, we build a recursive schema for canonical

i

functorial liftings of *H*@ over **Set**H@ . To do this, we define (*H*@)*∗* in cases and

j *i* j

expressly define (*H*@)*∗*(*A,* [ ]i)= (*H*@*A,* [ ]i). (Note that we use the same notation

j j

for the two i-ary operations). When *i* =1

* [(*L*j *a*)]1 = *L*j([ *a*]1)
* [(*B*j *t*1*...t*j)]1 = *B*j [*t*1]1*...*[*t*j]1

Likewise when *j* =1 we have

* [*L*1*a*1*, ...L*1*a*i]i = *L*1[*a*1*, ...a*i]i
* [*L*1*a*1*, ...L*1*a*i*−*1*,* (*B*1 *t*)]i = *B*1 [*L*1*a*1*, ...L*1*a*i*−*1*, t*]i
* *etc*
* [(*B*1 *t*1) *t*2*...t*i]i = *B*1 [*t*1*, t*2*...t*i]i

Otherwise for *i, j ≥* 2

* [*L*j*a*1*, ...L*j*a*i]i = *L*j[*a*1*, ...a*i]i
* [*L*j*a*1*, ...L*j*a*i*−*1*,* (*B*j *t*i,1*...t*i,j)]i = *B*j [*L*j*a*1*, ...L*j*a*i*−*1*, t*i,1]i *t*i,2*...t*i,j
* *etc*
* [(*B*j *t*1,1*...t*1,j) *t*2*...t*i]i = *B*j *t*1,1*...t*1,j*−*1 [*t*1,j*, t*2*...t*i]i

**Theorem 4.12** *For i, j ≥* 1*, there exists a schema of recursively deﬁned near dis-*

*tributive laws λ* : *H*@*H*@

*→ H*@*H*@

*between all free monads H*@*,H*@

*as deﬁned*

*above.*

i j j i i j

**Proof.** A near distributive law *λ* is created via the lifting functor (*H*@)*∗* over

j

*H*@ algebras described above. Applying (*H*@)*∗* to (*H*@*A, B*i), the *i*-ary operation

i j i

associated to the canonical algebra (*H*@*A, μ*) generates *λ* defined by the following

i

set of equations:

* + *λ*(*L*i*L*j *a*)= *L*j*L*i *a*
  + *λL*i(*B*j*t*1*... t*j)= *B*j(*λ L*i*t*1)*...*(*λ L*i*t*j)
  + *λ*(*B*i *t*1*... t*i)= [*λt*i]i where [ ]i is defined as in the previous result

Verifying that the two laws (DL C) and (DL D) hold follows from a straightforward argument via structural recursion and is left to the reader.

*2*

**Example 4.13** For the special case of *i* = 1 of Theorem [4.12](#_bookmark12), *H*@ is the writer monad *N ×* where *N* is the commutative monoid of natural numbers *{*0*,* 1*,* 2*,.. .}* under addition. *λ* : *N × H*@ *a → H*@(*N × A*) is actually a distributive law where

1

j j

*λ*(*n, t*) distributes *n* to each leaf of tree *t*. Likewise for the special case of *j* = 1,

*λ* : *H*@(*N × A*) *→ N × H*@*A* is again a full distributive law as follows. For an

i i

arbitrary tree *t* in *H*@(*N × A*), *λ t* = (*k, t∗*) where *t∗* is the tree in *H*@*A*, with the

i i

same shape as *t*, generated by replacing every leaf in *t* of the form *L*i(*m, a*) by *L*i*a*

and where *k* equals the sum of all the *m*’s found in the leaves.

Are the near distributive laws of Theorem [4.12](#_bookmark12) always distributive laws as in the two cases of the previous example? The answer is no in every other case.

**Theorem 4.14** *For any i, j ≥* 2 *the near distributive law λ* : *H*@*H*@ *→ H*@*H*@

i j j i

*of Theorem* [*4.12*](#_bookmark12) *fails to produce a distributive law as one can produce a tree t ∈*

*H*@*H*@*H*@ *for which law (DL B) fails.*

i j j

**Proof.** For *λ* : *H*@*H*@ *→ H*@*H*@ we produce a tree *t ∈ H*@*H*@*H*@ with 4(*j −* 1) + *i*

i j j i i j j

leaves for which (DL B) fails. Let

* + *lt* = *B*j (*L*j (*L*j *a*1))*...*(*L*j (*L*j *a*j*−*1)) (*L*j(*B*j(*L*j*a*j)*...*(*L*j*a*2j*−*1)))
  + *rt* = *B*j (*L*j(*B*j(*L*j*a*2j)*...*(*L*j*a*3j*−*1))) (*L*j (*L*j *a*3j))*...*(*L*j (*L*j *a*4j*−*2))
  + *t* = *B*i(*L*i(*lt*)) (*L*i(*L*j(*L*j*b*1))) *...* (*L*i(*L*j(*L*j*b*i*−*2))) (*L*i(*rt*))

then (DL B) fails for this *t*. The details are left to the reader. *2*

# 5 Pre-Monads

**Definition 5.1** A **pre-monad** in *V* is **H** = (*H, μ, η*) with *H* : *V → V* a functor and with *η* : id *→ H*, *μ* : *HH → H* natural transformations.

Composition of pre-monads is easily obtained. If (*H, μ, η*) and (*K, ν, ρ*) are pre- monads and if *λ* : *HK → KH* is a natural transformation, we obtain the **composite** pre-monad

KλH νμ ρη

(*KH, KHKH −−−−−−→ KKHH −−−−→ KH,* id *−−−−→ KH*)

It develops that *λ* with additional axioms will classify a functorial lift of *K* through *V***H**. To make sense of this we will have to define the “Eilenberg-Moore” category *V***H**.

**Definition 5.2** The axioms defining an **algebra** (*X, ξ*) for a pre-monad **H** = (*H, μ, η*) and an **H**-homomorphism *f* : (*X, ξ*) *→* (*Y, θ*) are exactly the same as for a monad, namely

*X η*X )*HX* ( *μ*X

*HHX*

*HX Hf* ) *HY*

id *ξ*

*Hξ ξ θ*

X

v v v v

*X* ( *HX*

*ξ*

*X*  ) *Y*

*f*

As for monads, the resulting category of algebras is denoted *V***H**. It is well known that the theory of algebras for a monad provides an approach to developing universal algebra [[9](#_bookmark30)]. It is very frequently the case that for a pre-monad **K** there is an isomorphism Φ : *V***K** *→ V***K***•* with **K***•* a monad, i.e. that *V***K** *→V* is monadic. By the well-known “Beck tripleableness theorem” it is enough that *V* is complete and that *V***K** *→V* satisfies the solution set condition, since the coequalizer condition of the theorem always holds. This shows that pre-monads play a role in developing universal algebra. This idea will be developed elsewhere, but we present an example now, with emphasis on the idea that pre-monads may reduce complications for the programmer.

A **band** is a semigroup in which every element is idempotent. Bands arise as the variety of semigroups satisfying the additional equation *xx* = *x*, so the monad **B** for bands is a quotient monad *θ* : **L** *→* **B** of the monad of non-empty lists. Note that throughout the paper the list monad will refer to non-empty lists. From the point of view of the monad programmer, the construction of the free band *BX* (despite the fact that *BX* is finite when *X* is, unlike the situation with lists) is not intuitive. An entire section of [[6](#_bookmark28)] is devoted to the word problem involved.

We next introduce an approach to bands which uses only the list data type. This illustrates our claim that, by relaxing axioms, we can sometimes describe what we need using readily available data types. We shall see in Theorem [6.2](#_bookmark20) below how this result leads to a simplification in specifying near distributive laws. In effect, we have gotten around the work to solve the word problem for free bands by simply not needing it!

**Proposition 5.3** *Let* (*L, m, e*) *be the monad of non-empty lists in* **Set***. Modify this to the pre-monad* (*L, m, e*ˆ) *where e*ˆX *x* = [*x, x*]*. Then* **Set**(L,m,eˆ) *is the category of*

*bands.*

**Proof.** An algebra (*X, ξ*) satisfies

*X e*ˆX

id

)*LX* ( *m*X

*ξ*

*LLX*

*Lξ*

X

v v

*X* ( *LX*

*ξ*

We have

*x* = *ξ*[*x, x*] = *ξ*(*m*X[[*x*]*,* [*x*]]) = *ξ*(*Lξ*)[[*x*]*,* [*x*]])

= *ξ*([*ξ*[*x*]*, ξ*[*x*]] = *ξe*ˆX(*ξ*[*x*]) = *ξ*[*x*]

But then (*X, ξ*) is also an algebra of the list monad, that is a semigroup (*X,·*) with *ξ*([*x*1*,..., x*n]) = *x*1 *··· x*n. This semigroup is a band because *x*2 = *ξ*[*x, x*]= *x*. The remaining details are routine. *2*

Although every monad is a pre-monad, a pre-monad need not satisfy any of the three monad axioms. We do have two pre-monad laws or axioms (PME.1, PME.2) where “PM” stands for “pre-monad”.

**Proposition 5.4** *Pre-monads may be equivalently described as* (*H,* (*·*)#*, η*) *where*

f

*H* : *V → V is a functor, η* : *id → H is a natural transformation and X −−→ HY '→*

f#

*HX −−→ HY is an operator subject to the axioms*

**(PME.1)** For *g* : *Y → HZ, g*#

Hg (id*HZ* )

= *HY −−→ HHZ −−−−−−→ HZ*

#

**(PME.2)** For *f* : *X → HY* , *g* : *Y → Z,* (*Hg*) *f* # = ((*Hg*)*f* )#

As for monads, the correspondences are

# Hf μ*Y*

1. *f* = *HX −−→ HHY −−→ HY*
2. *μ*X = (idHX)#

**Proof.** Let (*H, μ, η*) be a pre-monad. For *f* : *X → HY* define *f* # : *HX → HY* as in ([4](#_bookmark15)). Since *H*(idHX) = idHX, (idHX)# = *μ*X, and this gives (PME.1). For (PME.2), let *f* : *X → HY* , *g* : *Y → Z*. Then

(*Hg*)*f* # = (*Hg*) *μ*Y (*Hf* )= *μ*Z (*HHg*) (*Hf* ) (*μ* natural)

= *μ*Z *H*((*Hg*)*f* )= ((*Hg*)*f* )#

Conversely, let (PME.1, PME.2) hold and define *μ* by ([5](#_bookmark15)). ([4](#_bookmark15)) holds by PME.1. For

*g* : *Y → Z*,

(*Hg*) *μ*Y = (*Hg*) (idHY )# = ( (*Hg*) idHY )# (PME.2)

= (*Hg*)# = *μ*Z (*HHg*)

which shows that *μ* is natural. To complete the proof, we show the two passages are inverse bijections. Start with (*·*)#, define *μ*Z = (idHZ)# and then (*·*)## by ([4](#_bookmark15)). Then (*·*)# = (*·*)## by (PME.1). Starting with *μ*, define (*·*)# as in ([4](#_bookmark15)) and then *ν*Z = (idHZ)#. Then *ν*Z = *μ*Z as is clear from setting *g* = idHZ in ([4](#_bookmark15)). The proof is complete. *2*

**Definition 5.5** Let **H** = (*H, μ, η*), **K** = (*K, ν, ρ*) be pre-monads. A **pre-monad map** *σ* : **H** *→* **K** is a natural transformation *σ* : *H → K* such that

id *η* ) *H* ( *μ HH*

*ρ* *σσ*

*σ*

v v

*K* ( *ν KK*

The definition is the same as the usual one for monad maps so that monads form a full subcategory of pre-monads.

**Definition 5.6** Given a pre-monad **H** in *V*, a **monad approximation** of **H** is a reflection *σ* : **H** *→* **K** of **H** in the full subcategory of monads.

**Theorem 5.7** *Let* **H** = (*H, μ, η*)*,* **K** = (*K, ν, ρ*) *be pre-monads in V. Then a pre- monad map σ* : *H → K induces a functor W* : *V***K** *→ V***H** *over V deﬁned by*

σ*X* ξ

1. *W* (*X, ξ*)= (*X, HX −−→ KX −−→ X*)

*If, additionally,* **K** *is a monad, then σ '→ W is bijective with inverse*

Hρ*X* γ*X*

1. *σ*X = *HX −−−−→ HKX −−−−→ KX*

*where* (*KX, γ*X)= *W* (*KX, ν*X)*.*

**Proof.** Given *σ*, we first show *W* (*X, ξ*) is an **H**-algebra. This follows from

*ξ σ*X *η*X = *ξ ρ*X = *id*X and

*ξ σ*X *μ*X = *ξ ν*X(*σσ*)X (*σ* pre-monad map)

= *ξ ν*X *σ*KX(*Hσ*X)

= *ξ*(*Kξ*)*σ*KX(*Hσ*X) (**K**-algebra)

= *ξ σ*X(*Hξ*)(*Hσ*X) (*σ* natural)

= *ξ σ*X *H*(*ξσ*X)

That *W* maps homomorphisms to homomorphisms is clear from the naturality of *σ*. Now assume that **K** is a monad, so that (*KX, ν*X) isa **K**-algebra. We next show that if *W '→ σ '→ W* then *W* = *W* . (Of course we cannot assume here that *σ* is a pre- monad map since that has not yet been shown). Starting with a **K**-algebra (*X, ξ*),

*W* (*X, ξ*) is (*X, HX*

σ

*−−→ KX*

*X*

ξ

*−−→ X*) where *σ*X = *HX*

Hρ*X*

*−−→ HKX*

γ*X*

*−−→ KX*

and (*KX, γ*X) = *W* (*KX, ν*X). As *ξ* : (*KX, ν*X) *→* (*X, ξ*) is a **K**-homomorphism,

*ξ* : (*KX, γ*X) *→ W* (*X, ξ*) is an **H**-homomorphism. Writing *W* (*X, ξ*) as (*X, δ*), we

have the commutative diagram

*Hρ*X

*HX*

id

) *HKX*

*Hξ*

*γ*X ) *KX*

*ξ*

X

v v

*HX*  ) *X*

*δ*

where the square is because *ξ* is a homomorphism and the triangle is a **K**-algebra axiom. But the top row is *σ*X, so *δ* = *ξ σ*X and *W* (*X, ξ*)= (*X, δ*)= *W* (*X, ξ*). We may apply this, in particular, to the **K**-algebra (*KX, ν*X) to establish that

1. *γ*X = *ν*X *σ*KX

We turn to showing that *W '→ σ* is well defined. For *f* : *X → Y* in *V*, *Kf* : (*KX, ν*X) *→* (*KY, ν*Y ) isa **K**-homomorphism. Applying *W* gives the square on the right in the diagram

*HX Hρ*X ) *HKX γ*X ) *KX*

*Hf HKf Kf*

v v v

) )

*HY Hρ*Y *HKY γ*Y *KY*

But the square on the left commutes because *ρ* is natural. Since the rows are *σ*X and *σ*Y , the perimeter of the diagram then shows that *σ* is natural. The first pre-monad map law is shown by

*σ*X *η*X = *γ*X(*Hρ*X)*η*X = *γ*X *η*KX *ρ*X (*η* natural)

= *ρ*X ( (*KX, γ*X) algebra) For the second pre-monad map law,

*ν*X(*σσ*)X = *ν*X *σ*KX(*Hσ*X) = *γ*X(*Hσ*X) (by ([8](#_bookmark16)) )

= *γ*X(*Hγ*X)(*HHρ*X) = *γ*X *μ*KX(*HHρ*X) ( (*KX, γ*) algebra)

= *γ*X(*Hρ*X)*μ*X (*μ* natural)

= *σ*X *μ*X

Finally, we show that if *σ '→ W '→ σ* then *σ* = *σ*. *σ*X = *γ*X(*Hρ*X) = *ν*X *σ*KX(*Hρ*X) ( by ([8](#_bookmark16)))

= *ν*X(*Kρ*X)*σ*X (*σ* natural)

= *σ*X (**K** monad)

*2*

**Theorem 5.8** *Let* **H** *be a pre-monad such that U* : *V***H** *→ V is monadic so that there exists a monad* **K** *and an isomorphism of categories* Φ : *V***K** *→ V***H** *over V. Then the corresponding pre-monad map σ* : **H** *→* **K** *of Theorem* [*5.7*](#_bookmark16) *is a monad approximation of* **H***.*

**Proof.** Let *α* : **H** *→* **M** be a pre-monad map with **M** a monad and show that there exists a unique monad map *β* completing the triangle below.

*H*  *σ* ) *K*

*α* *β*

v

*M*

Let *W* : *V***M** *→ V***H** correspond to *α* and let *β* be the unique monad map

*−−→ V*

corresponding to *V***M**

*−−*W*→ V***H** Φ*−*1

**K**. We leave the remaining details to the

reader. *2*

**Example 5.9** By the theorem, *σ* : (*L, m, e*ˆ) *→* **B** is a monad approximation of the band monad. This example can be easily generalized. Let (*L, m, e*ˆ) be the list premonad where *e*ˆ(*x*) = [*x, x, x,* *, x*] *n*-times and *m* is the usual counit for lists.

Then **Set**(L,m,eˆ) is the category of *n*-bands (semigroups in which *x*n = *x* for every element).

**Example 5.10** Let (*L, m*ˆ *, e*) be the list premonad where *e*(*x*) = [*x*] and

*m*ˆ [*l*1*,* *l*k]= *l*++ *l*++ *l k*-times where *l* has the same length as list *l*1 and all of its

elements are the first element of *l*1. Then *σ* : *L → N × A* where *σ* [*a*1 *a*k]= (*k, a*1)

has as monad approximation the reader monad *N × A* where *N* = (*N, ∗,* 1) is the monoid of natural numbers under multiplication.

**Example 5.11** A rectangular band is a semigroup in which every element is idem- potent and additionally the equation *xyz* = *xz* holds. Let (*L, m*ˆ *, e*ˆ) be the list

premonad with *e*ˆ(*x*)= [*x, x*] and *m*ˆ *ll* = [*fst fst ll, lst lst ll*] where *fst* and *lst* pick

out the first and last elements of a non-empty list. As in the case of Proposition [5.3](#_bookmark14), an algebra (*X, ξ*) in **Set**(L,mˆ,eˆ) satisfies *x* = *ξ*[*x*]. Additionally,

*x*(*yz*)= *ξ*[*ξ*[*x*]*, ξ*[*y, z*]]

= *ξ ◦ m*ˆ [[*x*]*,* [*y, z*]] = *ξ*[*x, z*]= *xz*

Similarly, (*xy*)*z* = *xz*. The rectangular band monad **R** = (*R, μ, η*) where *R A* = *A×A* is the monad approximation of (*L, m*ˆ *, e*ˆ) defined by the reflection *σ* [*x*]= (*x, x*) and *σ* [*x*1*,... x*n] = (*x*1*, x*n). One can easily check that *σ* is a premonad map and that the monad properties of (*R, μ, η*) can be derived directly from (*L, m*ˆ *, e*ˆ). See Example [6.5](#_bookmark23).

**Example 5.12** Let (*L, m*ˆ *, e*) be the list premonad where *e*(*x*) = [*x*], *m*ˆ [[*x*]] = [*x*] and *m*ˆ [*l*1*, ...l*k] = [*fst l*1*,fst l*1]. Then *σ* : *L → B × A* defined by *σ* [*a*] = (*F, a*), *σ* [*a*1*...a*k] = (*T, a*1) defines a monad approximation of the *Boolean*-set monad *B × A* where Bool= (*T, F* ), *B* = (Bool*, OR,F* ). The algebras (*X, ξ*) in **Set**(L,mˆ,e) are semigroups that satisfy *xy* = *x*2 since

*xy* = *ξ*[*x, y*]= *ξ*[*ξ*[*x*]*, ξ*[*y*]]

= *ξ ◦ m*ˆ [[*x*]*,* [*y*]] = *ξ*[*x, x*]= *x*2

# 6 Near Distributive Laws for Pre-Monads

We note that the laws DL A, DL B, DL C, DL D make sense whenever (*H, μ, η*), (*K, ν, ρ*) are pre-monads. In this section, we show how distributive laws and near distributive laws for pre-monads induce similar laws on their monad approximations. As shown in [[10](#_bookmark31), Theorem 2.2.2] the following result is well known when **H** is a monad. The generalization to the case when **H** is a pre-monad must be proved with some care.

**Theorem 6.1** *Let K* : *V → V be a functor,* (*M, m, e*) *a monad in V and let* (*H, μ, η*) *be a pre-monad in V such that V***H** *→V is monadic. Functorial lifts K*٨ : *V***M** *→ V***H** *correspond bijectively to natural transformations λ* : *HK → KM which satisfy* (*K*٨ *A, K*٨ *B*)*:*

*ηK*

*K*

(*K*٨ *A*)

) *HK*

( *μK*

(*K*٨ *B*)

*HHK*

ν*Hλ*

*λ*

*Ke*

ν

*HKM*

ν*λM*

The correspondences are

*KM* ( *KMM*

*Km*

٨ θ λ*X* Kθ

1. *K* (*X, MX −−→ X*)= (*KX, HKX −−→ KMX −−→ KX*)

and, if *K*٨(*MX, m*X)= (*KMX, γ*X),

HKe*X* γ*X*

1. *λ*X = *HKX −−−−→ HKMX −−−−→ KMX*

Moreover, half of this result holds if **M** is only a pre-monad, namely if *λ* satisfies (*K*٨ *A*) and (*K*٨ *B*), then *K*٨ as in ([9](#_bookmark19)) is a functorial lift *V***M** *→ V***H** of *K*.

**Proof.** Given *λ* satisfying (*K*٨ *A*) and (*K*٨ *B*) and assuming that **M**, **H** are arbi- trary pre-monads, we first show that (*KMX,* (*Kθ*) *λ*X) is an **H**-algebra by checking the properties in Definition [5.2](#_bookmark13). We have

(*Kθ*)*λ*X *η*KX = (*Kθ*)(*Ke*X) ((*K∗A*) for *λ*)

= *id*KX (*θ* algebra)

Also,

(*Kθ*)*λ*X *μ*KX = (*Kθ*)(*Km*X)*λ*MX(*Hλ*X) ((K*∗* B) for *λ*)

= (*Kθ*)(*KMθ*)*λ*MX(*Hλ*X) (*θ* algebra)

= (*Kθ*)*λ*X(*HKθ*)(*Hλ*X) (*λ* natural)

Further an **M**-homomorphism *f* : (*X, ξ*) *→* (*Y, θ*), *Kf* is an **H**-homomorphism as follows:

(*Kf* )(*Kξ*)*λ*X = (*Kθ*)(*KMf* )*λ*X (**M**-homomorphism)

= (*Kθ*)*λ*Y (*HKf* ) (*λ* natural)

Conversely, now assuming that **M** is a monad, let *K*٨ : *V***M** *→ V***H** be a functorial

lift of *K*. Let *σ* : *H → H*^

be such that (*H*^ *, μ*^*, η*^) is the monad approximation

corresponding to the isomorphism Φ : *V***H**^ *→ V***H**, (*X, ξ*) *'→* (*X, ξσ* ) as in Theorem

X

[5.8](#_bookmark17). This gives rise to a new functorial lift

*−−−−→V*

*V***M** K*s*

*−−−−→V*

**H** Φ*−*1 **H**^

of *K* which, since the theorem holds for monads, corresponds to a natural transfor- mation *λ*^ : *H*^ *K → KM* which satisfies (*K*٨ *A*) and (*K*٨ *B*). Define

σK λ^

*λ* = *HK −−−−→ H*^ *K −−−−→ KM*

We first show that such *λ* satisfies (K*∗* A) and (K*∗* B).

*λ*^(*σK*)(*ηK*)= *λ*^(*η*^*K*) (*σ* pre-monad map)

= *Ke* ((*K∗* A) for *λ*^)

*λ*^(*σK*)(*μK*)= *λ*^(*μ*^*K*)(*σH*^ *K*)(*HσK*) (*σ* pre-monad map)

= (*Km*)(*λ*^*M* )(*H*^ *λ*^)(*σH*^ *K*)(*HσK*) ((*K*٨ *B*) for *λ*^)

= (*Km*)(*λ*^*M* )(*σKM* )(*Hλ*^)(*HσK*) (*σ* natural)

as desired.

If *K*٨(*MX, m*X) = (*KMX, γ*X) there exists a unique **H**^ -algebra (*KMX, γ*^X)

γˆ*X*

with *γ*X = *HKMX −−−→ H*^ *KMX −−→ KMX*. By ([10](#_bookmark19)), which holds since **H**^ is a

σ*nMX*

H^ Ke*X*

^ ^ ^

γ^*X*

monad, *λ*X = *HKM X −−−−→ HKM X −−→ KMX*. We can then check that *λ* is

also defined by ([10](#_bookmark19)) as follows.

*λ*^X *σ*KX = *γ*X (*H*^ *Ke*X) *σ*KX

^

= *γ*X *σ*KMX (*HKe*X) (*σ* natural)

^

= *γ*X (*HKe*X)

So far, the passages of ([9,](#_bookmark19) [10](#_bookmark19)) are well defined. If *λ '→ K*٨ *'→ λ*1 then *γ*X = (*Km*X) *λ*MX so

*λ*1,X = (*Km*X) *λ*MX (*HKe*X) = (*Km*X) (*KMe*X) *λ*X (*λ* natural)

= *id*KMX *λ*X (*m*X algebra)

= *λ*X

If *K*٨ *'→ λ '→ K•*, let *K*٨(*X, ξ*) = (*KX, HKX*

*−−*δ*→ KX*). Taking *K*٨ of the

**M**-homomorphism *ξ* : (*MX, m*X) *→* (*X, ξ*) gives a commutative square

*HKMX*

*HKξ*

) *HKX*

*γ*X *δ*

v v

)

*KMX Kξ KX*

Then *K•*(*X, ξ*)= (*KX, ϵ*) where

*ϵ* = (*Kξ*) *λ*X = (*Kξ*) *γ*X (*KHe*X)

= *δ* (*HK*(*ξe*X)) = *δ* (*ξ* algebra)

*2*

We conclude the section with the promised theorems and point out the connec- tion to Proposition [5.3](#_bookmark14) (which asserts that the band monad is the monad approxi- mation of (*L, m, e*^) ).

**Theorem 6.2** *Let* **K** = (*K, ν, ρ*) *be a pre-monad in V and let* **H** = (*H, μ, η*) *be*

*a pre-monad in V with monad approximation σ* : **H** *→* **H**^ *,* **H**^ = (*H*^ *, μ*^*, η*^)*. Let*

*λ* : *HK → KH be a natural transformation satisﬁng (DL C, DL D). Then there*

*exists a near distributive law λ*^ : *H*^ *K → KH*^

*square commutes. We say λ* ***generates*** *λ*^*.*

*σK* )

*of* **H**^

*over* **K** *such that the following*

*HK*

*λ*

v

)

*KH Kσ*

*H*^ *K*

*λ*

^

v

*KH*^

([6](#_bookmark20))

**Proof.** For *σ* : *H → H*^ the monad approximation of *H*, the isomorphism Φ : *V***H**^ *→*

*V***H** maps the free algebra (*H*^ *X, μ*^

*X*

) to the **H**-algebra (*H*^ *X, μ*^*X*

*σ*H^ X

), *K*٨ of which

is (*KH*^ *X, ω*X) where

λ*H*ˆ *X*

Kσ*H*ˆ *X*

Kμˆ*X*

*ω*X = *HKH*^ *X −−−−→ KHH*^ *X −−−−→ KH*^ *H*^ *X −−−−→ KH*ˆ *X*

Applying Theorem 6.1 to the composite functorial lift through *V*H^ , (*H*^ *X, μ*^ ) is

X

^ ^ ^ ^

mapped to (*KHX, Kη*), the **H**-algebra corresponding to the **H**-algebra (*KHX, ω*X). This lift is classified by

H^ Kη^*X* ω*•*

*λ*^X = *H*^ *KX −−−−→ H*^ *KH*^ *X −−−X−→ KH*^ *X*

Then *λ* satisfies (DL C, DL D) because it arises from the formula corresponding to a functorial lift. To complete the proof, we must show that the square ([6](#_bookmark20)) commutes. We have

^

*λ*^ (*σK*)= *ω•* (*H*^ *Kη*^) (*σK*) = *ω•*(*σKH*^ ) (*HKη*^) (*σ* natural)

= *ω* (*HKη*^) (by the definition of *ω•*)

= (*Kμ*^) (*KσH*^ ) (*λH*^ )(*HKη*^) = (*Kμ*^)(*Kσ*) (*KHη*^) *λ* (*λ* natural)

= (*Kμ*^) (*KH*^ *η*^) (*Kσ*) *λ* (*σ* natural)

= (*Kσ*) *λ* (**H**^ -algebra)

*2*

**Lemma 6.3** *Let* **H** = (*H, η,* (*·*)#) *and* **K** = (*K, e,* (*·*)##) *be pre-monads in V. Given a map σ*X : *HX → KX (not necessarily assumed to be a natural trans- formation) for each X, σ is a pre-monad map if and only if ση* = *e and for each f* : *X → HY the following square commutes.*

*HX σ*X) *KX*

*f* # (*σ*Y *f* )## ([6*.*3](#_bookmark21))

v v

)

*HY σ*Y *KY*

**Proof.** We first show that if *σ* is a pre-monad map then ([6.3](#_bookmark21)) holds. *f* #

μ*Y*

*HHY −−→ HY* .

*σ*Y *f* # = *σ*Y *μ*Y (*Hf* ) = *μ*X(*Kσ*X)(*σ*HY )(*Hf* )

= *μ*Y (*Kσ*Y )(*Kf* ) *σ*X (*σ* is natural)

= (*σ*Y *f* )## *σ*X

Now the other direction. Let *g* : *X → Y* . Then (*Kg*) *σ*X = (*e*Y *g*)## *σ*X = (*σ*Y *η*Y *g*)##*σ*X

= *σ*Y (*η*Y *g*)# (by ([6.3](#_bookmark21)))

= *σ*Y (*Hg*)

shows that *σ* is natural. To see that *σ* is a pre-monad map,

*m*X (*σσ*)X = *m*X (*Kσ*X) *σ*HX = (*σ*X idHX)## *σ*HX

= *σ*X (idHX)# (by ([6.3](#_bookmark21)))

= *σ*X *μ*X

Hf

= *HX −−→*

*2*

**Proposition 6.4** *Let σ* : (*H, η,* (*·*)#) *→* (*K, e,* (*·*)##) *be a pre-monad map, let f* : *X → KY and let s* : *KX → HX, t* : *KY → HY be morphisms such that σ*X *s* = *id*KX *, σ*Y *t* = *id*KY *. Then*

*f* ## = *σ*Y (*tf* )# *s*

**Proof.** As *σ* is a pre-monad map, we have by Lemma [6.3](#_bookmark21)

*f* ## = (*σ*Y *tf* )## *σ*X *s* = *σ*Y (*tf* )# *s*

*2*

**Example 6.5** The calculation of *μ* for the rectangular band monad (*R, μ, η*) is not immediately intuitive. We can derive it however applying Proposition [6.4](#_bookmark22) where *R* is the monad approximation of *L* of Example [5.11](#_bookmark18) so that *σ*X[*x*1*,..., x*m]= (*x*1*, x*m). Define a splitting *s* : *R → L* by *s*(*x, y*)= [*x, y*], so *μ*X = (*id*RX)# = *σ ◦* (*s*X *◦ id*)# *◦ s*X = *σ*X *◦ m*ˆ *◦ L*(*s*X) *◦ s*X and thus *μ*(*a, b, c, d*)= *σ*X *◦ m*ˆ *◦ L*(*s*X) *◦ s*X(*a, b, c, d*)= *σ*X *◦ m*ˆ *◦ L*(*s*X)[(*a, b*)*,* (*c, d*)] = *σ*X *◦ m*ˆ [[*a, b*]*,* [*c, d*]] = *σ*X[*a, d*]= (*a, d*) as expected.

**Example 6.6** The previous example can be generalized to other dimensions. For *n ≥* 1, let (*L*n*, μ, η*) denote the monad of lists of length exactly *n*. For instance when *n* = 3, *η*(*a*) = [*a, a, a*] while *μ*([[*a, b, c*]*,* [*d, e, f* ]*,* [*g, h, i*]]) = [*a, e, i*] defines monad (*L*3*, μ, η*). *L*3 is the monad approximation associated with premonad (*L, m*ˆ *, e*ˆ) de- fined by *e*ˆ(*x*)= [*x, x, x*] and *m*ˆ (*ll*)= [*p*1(*p*1 *ll*)*, p*2(*p*2 *ll*)*, p*3(*p*3 *ll*)] where *p*i picks out the *i − th* element in a list(or the last element if the list is too small). *L*n, which is equivalent to the cartesian product of the identity monad (*n*-times), (*id*)n, is the monad approximation of a premonad structure on lists *L*, similar to the previous example.

When *V* = **Set**, the image of *σ* : *H → H*^ is a submonad with the universal

property. Thus all monad approximations are pointwise split epic in **Set**.

**Theorem 6.7** *If σ* : **H** *→* **H**^ *is a pointwise split epic monad approximation then if*

*λ* : *HK → KH is a distributive law then so too is λ*^ : *H*^ *K → KH*^ *.*

**Proof.** For (DL A),

*λ*^(*H*^ *ρ*)*σ* = *λ*^(*σ*K*Hρ*) (*σ* natural)

= (*Kσ*)*λ*(*Hρ*) by ([6](#_bookmark20))

= (*Kσ*)*ρ*H (*λ* a distributive law)

= *ρ*H^ *σ* (*ρ* natural)

so (DL A) holds as *σ* is pointwise epic. Similarly, for (DL B),

*λ*^(*H*^ *ν*)*σ*KK = *λ*^(*σ*K*Hν*) (*σ* natural)

= (*Kσ*)*λ*(*Hν*) by ([6](#_bookmark20))

= (*Kσ*)(*ν*H)(*Kλ*)(*λ*K) (DL B for *λ*)

= (*ν*H^ )(*KKσ*)(*Kλ*)(*λ*K) (*ν* natural)

= (*ν*H^ )(*Kλ*^)(*Kσ*K)*λ*K by ([6](#_bookmark20))

= (*ν*H^ )(*Kλ*^)*λ*^K *σ*KK by ([6](#_bookmark20))

Since *σ* is a retraction, it is surjective, so (DLA) and (DL B) hold for *λ*^. *2*

# References

1. H. Appelgate, *Acyclic Models and Resolvent Functors*, Dissertation, Mathematics Department, Columbia University, 1963.
2. M. Barr, Coequalizers and free triples, *Mathematische Zeitschrifte* 116, 1970, 307–322.
3. J. Beck, Distributive laws, *Lecture Notes in Mathematics* 80, Springer-Verlag, 1969, 119–140.
4. R. S. Bird and O. de Moor, *The Algebra of Programming*, Prentice-Hall, 1996.
5. N. Hindman and D.Strauss, *Algebra in the Stone-Cech Compactification: Theory and Applications*, De Gruyter, 1998.
6. J. M. Howie, *An Introduction to Semigroup Theory*, Academic Press, 1971.
7. P. T. Johnstone, Adjoint lifting theorems for categories and algebras, *Bulletin of the London* *Mathematical Society* 7, 1975, 294–297.
8. V. Kurkova-Pohlov´a and V. Koubek, When a generalized algebraic category is monadic,

*Commentationes Mathematicae Universitatis Carolinae* 15, 1974, 577–587.

1. E. G. Manes, Monads of sets, in M. Hazewinkel (ed.), *Handbook of Algebra*, Vol. 3, Elsevier Science B.V., 2003, 67–153.
2. E. G. Manes and P. Mulry, Monad compositions I: general constructions and recursive distributive laws, *Theory and Applications of Categories* 18, 2007, 172–208.
3. E. G. Manes and P. Mulry, Monad compositions II: Kleisli strength, *Mathematical Structures in Computer Science*18, 2008, 613–643.
4. P. Mulry, Notions of Monad Strength, *Semantics, Abstract Interpretation, and Reasoning about Programs, Electronic Proceedings in Theoretical Computer Science*, vol 129 2013, 67-83.