Monad-comonad interaction laws

Dylan McDermott Exequiel Rivas Tarmo Uustalu

ICFP 2021 tutorials, Daejon / online, 22–27 August 2021

What is this tutorial about?

- In FP, we are used to modelling effectful notions of computations with monads.
- But this alone does not capture how effectful computations are to be **run**!
- An effectful computation needs an environment to run against.
- We'll explain:
  - modelling notions of environment with comonads.
  - modelling how computations and environments can work together with **monad-comonad interaction laws**.

Prerequisites

- This is indispensable:
- FP knowledge about effectful computation and modelling of notions of effect with monads, as introduced by Moggi and customary in Haskell
- Basic category theory for FP

- We will also use need to introduce advanced category theory, but will try to convey the intuitions and significance for FP
- In particular, we demonstrate most of the material we introduce in Haskell.

- We will work with a fairly general base category, varying the assumptions).
- But all examples make sense in **Set**, which **Hask** approximates.

Outline

- First facts about (non-residual and residual) **monad-comonad interaction laws**
  - (non-residual) monad-comonad int laws
  - the category of monad-comonad int laws
  - residual monad-comonad int laws
- **Dual** of a comonad, **Sweedler** dual of a monad
  - the "greatest" interacting monad resp comonad for a given comonad or monad
- Monad-comonad interaction laws **(co)algebraically**
  - connection of int laws to handlers and cohandlers!
- **Monoid-comonoid interaction laws**, Sweedler theory of monads
  - raising the abstraction level to recognize and exploit a setting from abstract algebra

Monad-comonad interaction laws

Effects happen in interaction

- To run,

  an effectful (effect-requesting) program behaving as
  a computation

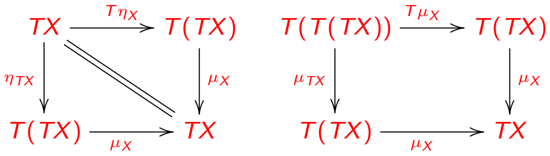
  needs to **interact** with

  a environment
  that an effect-providing (coeffectful) machine behaves as

- E.g.,
  - a nondeterministic program needs a machine making choices;
  - a stateful program needs a machine coherently responding to fetch and store commands.

Monads

- Given a category $\mathcal{C}$, eg **Set**.
- A *monad* on it is given by a a functor $T : \mathcal{C} \to \mathcal{C}$ and nat transfs $\eta : \mathsf{Id}_{\mathcal{C}} \to T$ (the *unit*), $\mu : T \cdot T \to T$ (the *multiplication*) such that



- Legend:
  T – a notion of computation
  X – values, TX – computations, $T(TX)$ – computations of computations,
  η_X – turns a value into a just-returning computation,
  μ_X – turns a computation of computations into their sequence
- A computation $\approx$ a process that can do various things, incl asking for outside help, and may eventually finish and return a value.

Some examples of monads

- Assume $\mathcal{C}$ is Cartesian closed (symm mon closed works too).
- Read-only state monad (reader monad) for S:

$$
\begin{aligned}
&TX = S \Rightarrow X \qquad \text{for some } S \\
&\eta\, x = \lambda\_.\, x \\
&\mu\, cc = \lambda s.\, (cc\, s)\, s
\end{aligned}
$$

- State monad:

$$
\begin{aligned}
&TX = S \Rightarrow (S \times X) \\
&\eta\, x = \lambda s.\, (s, x) \\
&\mu\, cc = \lambda s.\, \text{let } (s', c) = cc\, s \text{ in } c\, s'
\end{aligned}
$$

- Intensional state monad:

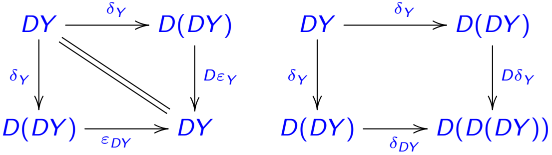
$$
\begin{aligned}
TX = {} & \mu Z.\, X + (S \Rightarrow Z) + (S \times Z) \\
& (TX =^{\mu} X + (S \Rightarrow TX) + (S \times TX)), \\
& \text{(wellfounded leaf-labelled trees)}
\end{aligned}
$$

T is a free functor-algebras monad (free monad).

- Legend: X – values, S – data states

Comonads

- A *comonad* on $\mathcal{C}$ is given by a functor $D : \mathcal{C} \to \mathcal{C}$ and nat transfs $\varepsilon : D \to \mathsf{Id}_{\mathcal{C}}$ (the *counit*), $\delta : D \to D \cdot D$ (the *comultiplication*) such that



- Legend:
  D – a notion of environments
  Y – states, DY – environments,
  $D(DY)$ – environments with, as states, environments,
  ε_Y – extracts from an environment its initial state
  (useful for halting),
  δ_Y – blows an envmt up into one with current remainders as states
  (useful for pausing/resuming)
- An environment $\approx$ a process that is able to serve computations and is at any moment in some state.

Some examples of comonads

- Read-only costate comonad (coreader comonad) for S:

$$
\begin{aligned}
&DY = S \times Y \\
&\varepsilon\,(\_, y) = y \\
&\delta\,(s, y) = (s, (s, y))
\end{aligned}
$$

- Costate comonad:

$$
\begin{aligned}
&DY = S \times (S \Rightarrow Y) \\
&\varepsilon\,(s, f) = f\ s \\
&\delta\,(s, f) = (s, \lambda s'.\,(s', f))
\end{aligned}
$$

- Intensional costate comonad:

$$
\begin{aligned}
DY = {} & \nu W.\, Y \times (S \times W) \times (S \Rightarrow W) \\
& (DY =^{\nu} Y \times (S \times DY) \times (S \Rightarrow DY)),
\end{aligned}
$$

(nonwellfounded node-labelled trees)

D is a cofree functor-coalgebras comonad (cofree comonad).

- Legend: Y – (control) states, S – data states

Functor-functor interaction laws

- Assume $\mathcal{C}$ is a Cartesian category. (Symmetric monoidal works too.)
- A *functor-functor interaction law*[1] is given by two functors $F, G : \mathcal{C} \to \mathcal{C}$ and a family of maps

$$\phi_{X,Y} : FX \times GY \to X \times Y$$

natural in X, Y.

- Legend:
  F – a notion of computations
  (without just-returning and sequencing)
  X – values, FX – computations

  G – a notion of environments
  (without halting and pausing/resuming)
  Y – states, GY – environments

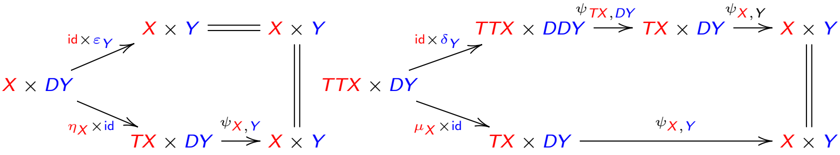
[1] a *pairing* in Kmett's terminology

Monad-comonad interaction laws

- A *monad-comonad interaction law* is given by a monad (T, η, μ) and a comonad (D, ε, δ) and a family of maps

$$\psi_{X,Y} : TX \times DY \to X \times Y$$

natural in X and Y such that



- Legend:
  T – a notion of computations
  X – values, TX – computations

  D – a notion of environments
  Y – states, DY – environments

Some exampes of mnd-comnd int laws: Read-only state (1)

- Assume $\mathcal{C}$ is Cartesian closed (or symm monoidal closed).
- $TX = S \Rightarrow X$ – the reader monad
  $\eta\, x = \lambda \_.\, x$
  $\mu\, cc = \lambda s.\, (cc\, s)\, s$

  $DY = S \times Y$ – the coreader comonad
  $\varepsilon\, (\_, y) = y$
  $\delta\, (s, y) = (s, (s, y))$

  for some S
- $\psi_{X,Y} : TX \times DY \to X \times Y$

  $\psi\, (c, (s, y)) = (c\, s, y)$
- Legend:
  X – values,
  Y – (control) states, S – data states

Read-only state (2)

- $TX = V \Rightarrow X$

  $DY = S \times Y$

  for some S, V and $view : S \to V$

- $\psi_{X,Y} : TX \times DY \to X \times Y$

  $\psi\,(c,(s,y)) = (c\,(view\,s),y)$

- Legend:
  X – values, V – "views" of data states,
  Y – (control) states, S – data states

State (1)

- $TX = S \Rightarrow (S \times X)$ – the state monad
  $\eta\, x = \lambda s.\, (s, x)$
  $\mu\, cc = \lambda s.\, \text{let } (s', c) = cc\, s \text{ in } c\, s'$

  $DY = S \times (S \Rightarrow Y)$ – the costate comonad
  $\varepsilon\, (s, f) = f\, s$
  $\delta\, (s, f) = (s, \lambda s'.\, (s', f))$

  for some S

- $\psi_{X,Y} : TX \times DY \to X \times Y$

  $\psi\, (c, (s, f)) = \text{let } (s', x) = c\, s \text{ in } (x, f\, s')$

- Legend:
  X – values,
  Y – (control) states, S – data states

State (2)

- $TX = V \Rightarrow (V \times X)$

  $DY = S \times (S \Rightarrow Y)$

  for some S, V, $\mathit{view} : S \to V$ and $\mathit{set} : S \times V \to S$ forming a *(very well-behaved) lens*

- $\psi_{X,Y} : TX \times DY \to X \times Y$

  $\psi\,(c, (s, f)) = \text{let } (v', x) = c\,(\mathit{view}\,s) \text{ in } (x, f\,(\mathit{set}\,(s, v')))$

- Legend:
  X – values, V – "views" of data states,
  Y – (control) states, S – data states

Read-only state vs costate

- $TX = V \Rightarrow X$

  $DY = S \times (S \Rightarrow Y)$

  for some S, V, $view : S \to V$ and $set : S \times V \to S$ forming a lens

- $\psi_{X,Y} : TX \times DY \to X \times Y$

  $\psi\,(c, (s, f)) = (c\,(view\,s), f\,s)$

Intensional state

- $TX = \mu Z.\, X + (S \Rightarrow Z) + (S \times Z)$
  $(TX =^{\mu} X + (S \Rightarrow TX) + (S \times TX))$
  – the intensional state monad for

  $DY = \nu W.\, Y \times (S \times W) \times (S \Rightarrow W)$
  $(DY =^{\nu} Y \times (S \times DY) \times (S \Rightarrow DY))$
  – the intensional costate comonad

  for some S

- $\psi_{X,Y} : TX \times DY \to X \times Y$

  $\psi\,(\mathrm{in}\,(\mathrm{inl}\,x), e) = (x, \mathrm{fst}\,(\mathrm{out}\,e))$

  $\psi\,(\mathrm{in}\,(\mathrm{inr}\,(\mathrm{inl}\,f)), e) = \mathrm{let}\ (s, e') = \mathrm{fst}\,(\mathrm{snd}\,(\mathrm{out}\,e))\ \mathrm{in}\ \psi\,(f\,s, e')$
  $\psi\,(\mathrm{in}\,(\mathrm{inr}\,(\mathrm{inr}\,(s, c))), e) = \psi\,(c, \mathrm{snd}\,(\mathrm{snd}\,(\mathrm{out}\,e))\,s)$

Intensional state vs extensional costate

- $TX = \mu Z.\, X + (S \Rightarrow Z) + (S \times Z)$,
  $DY = S \times (S \Rightarrow Y)$
- $\psi_{X,Y} : TX \times DY \to X \times Y$

  $\psi\,(\mathsf{in}\,(\mathsf{inl}\,x), (s, g)) = (x, g\,s)$

  $\psi\,(\mathsf{in}\,(\mathsf{inr}\,(\mathsf{inl}\,f)), (s, g)) = \psi\,(f\,s, (s, g))$
  $\psi\,(\mathsf{in}\,(\mathsf{inr}\,(\mathsf{inr}\,(s', c))), (s, g)) = \psi\,(c, (s', g))$

Intensional nondeterminism

- $TX = \mu Z.\, X + Z \times Z$
  $DY = \nu W.\, Y \times (W + W)$
- $\psi_{X,Y} : TX \times DY \to X \times Y$
  $\psi\,(\mathsf{in}\,(\mathsf{inl}\,x), e) = (x, \mathsf{fst}\,(\mathsf{out}\,e))$
  $\psi\,(\mathsf{in}\,(\mathsf{inr}\,(c_0, c_1)), e) = \mathsf{case}\ \mathsf{snd}\,(\mathsf{out}\,e)\ \mathsf{of} \begin{cases} \mathsf{inl}\,e' \mapsto \psi\,(c_0, e') \\ \mathsf{inr}\,e' \mapsto \psi\,(c_1, e') \end{cases}$
- T is a free functor-algebras monad (free monad).
  D is a cofree functor-coalgebras comonad (cofree comonad).

Int law maps

- A *functor-functor interaction law map* between (F, G, ϕ), (F', G', ϕ') is given by nat. transfs. $f : F \to F'$, $g : G' \to G$ such that

$$
\begin{array}{ccccc}
 & & FX \times GY & \xrightarrow{\phi_{X,Y}} & X \times Y \\
 & \overset{\mathrm{id} \times g_Y}{\nearrow} & & & \Vert \\
FX \times G'Y & & & & \\
 & \underset{f_X \times \mathrm{id}}{\searrow} & & & \\
 & & F'X \times G'Y & \xrightarrow{\phi'_{X,Y}} & X \times Y
\end{array}
$$

- Functor-functor interaction laws form a category.

- A *monad-comonad interaction law map* between (T, D, ψ) and (T', D', ψ') is given by a monad map $f : T \to T'$ and a comonad map $g : D' \to D$ such that

$$
\begin{array}{ccccc}
 & & TX \times DY & \xrightarrow{\psi_{X,Y}} & X \times Y \\
 & \overset{\mathrm{id} \times g_Y}{\nearrow} & & & \Vert \\
TX \times D'Y & & & & \\
 & \underset{f_X \times \mathrm{id}}{\searrow} & & & \\
 & & T'X \times D'Y & \xrightarrow{\psi'_{X,Y}} & X \times Y
\end{array}
$$

- Monad-comonad interaction laws also form a category.

Monad-comonad int laws as monoids

- The category of functor-functor int laws has a composition-based monoidal structure.

- The unit is $(\mathsf{Id}, \mathsf{Id}, \phi)$ where $\phi_{X,Y} = \mathrm{id}_{X \times Y}$.
- The composition of (F_0, G_0, ϕ_0) and (F_1, G_1, ϕ_1) is $(F_0 \cdot F_1, G_0 \cdot G_1, \phi)$ where

$$\phi_{X,Y} = F_0(F_1 X) \times G_0(G_1 Y) \xrightarrow{\phi_{0\,F_1 X, G_1 Y}} F_1 X \times G_1 Y \xrightarrow{\phi_{1\,X,Y}} X \times Y$$

- These categories are isomorphic:
  - monad-comonad int laws
  - monoids in the category of functor-functor int laws

Some constructions with mnd-cmnd int laws: Stretching

- Given a monad-comonad int law ψ of T, D, pre-composing with a monad map $f : T' \to T$ and a comonad map $g : D' \to D$, gives a monad-comonad int law ψ' of T', D' by

$$\psi'_{X,Y} = \; T'X \times D'Y \xrightarrow{f_X \times g_Y} TX \times DY \xrightarrow{\psi_{X,Y}} X \times Y$$

- From reader to state:

$$\begin{array}{l} T'X = S \Rightarrow X \\ TX = S \Rightarrow (S \times X) \\ f_X : T'X \to TX \\ f\,c = \lambda s.\,(s, c\,s) \end{array}$$

- From state to state:

$$\begin{array}{l} TX = V \Rightarrow (V \times X) \\ T'X = S \Rightarrow (S \times X) \\ f_X : T'X \to TX \\ f\,c = \lambda s.\, \mathit{let}\ (v', x) = \mathit{view}\,(c\,s)\ \text{in}\ (\mathit{set}\,(s, v'), x) \end{array}$$

Initial and final monad-comonad int laws

- The initial monad-comonad interaction law is (T, D, ψ) where $TX = X$, $DY = Y$ and

$$\psi_{X,Y} = \; X \times Y = X \times Y$$

- If $\mathcal{C}$ is distributive, the final monad-comonad interaction law is (T, D, ψ) where $TX = 1$, $DX = 0$ and

$$\psi_{X,Y} = \; 1 \times 0 \longrightarrow 0 \xrightarrow{?} X \times Y$$

Product of two mnd-comnd int laws

- If $\mathcal{C}$ is distributive, the product of two mnd-comnd int laws (T_0, D_0, ψ_0) and (T_1, D_1, ψ_1) is (T, D, ψ) where

  $TX = T_0X \times T_1X$,
  $DY = D_0Y + D_1Y$ and

$$\psi_{X,Y} = \begin{array}{c} (T_0X \times T_1X) \times (D_0Y + D_1Y) \\ \downarrow \\ (T_0X \times D_0Y) + (T_1X \times D_1Y) \\ \downarrow [\psi_{0X,Y}, \psi_{1X,Y}] \\ X \times Y \end{array}$$

Free mnd-comnd int law on a func-func int law

- If $\mathcal{C}$ is Cartesian closed (or symm mon closed) and the relevant initial algebras and final coalgebras exist,
  the free mnd-comnd int law on a func-func int law (F, G, ϕ) is (T, D, ψ) where

  $TX = F^* X = \mu Z.\, X + FZ$ – the free monad on F –,
  $DY = G^\dagger Y = \nu W.\, Y \times GW$ – the cofree comonad on G –,

  and the currying of $\psi_{X,Y}$ is induced by the currying of the map

$$
\begin{array}{c}
(X + F(DY \Rightarrow X \times Y)) \times DY \\
\downarrow \scriptstyle{\mathrm{id}+\mathrm{out}} \\
(X + F(DY \Rightarrow X \times Y)) \times (Y \times G(DY)) \\
\downarrow \\
X \times Y + F(DY \Rightarrow X \times Y) \times G(DY) \\
\downarrow \scriptstyle{\mathrm{id}+\phi_{DY \Rightarrow X \times Y, DY}} \\
X \times Y + (DY \Rightarrow X \times Y) \times DY \\
\downarrow \scriptstyle{\mathrm{id}+\mathrm{ev}} \\
X \times Y + X \times Y \\
\downarrow \\
X \times Y
\end{array}
$$

Degeneracies!

- The functor Maybe is difficult to interact with!
- An interaction law ϕ with G would give, for any Y, a map $GY \to 0$:

$$GY \longrightarrow 1 \times GY \xrightarrow{\text{nothing} \times \text{id}} \text{Maybe}\, 0 \times GY \xrightarrow{\phi_{0,Y}} 0 \times Y \xrightarrow{\text{fst}} 0$$

- There are also other problematic functors and monads.

Some degeneracy thms for func-func int laws

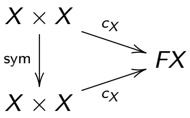
- Assume $\mathcal{C}$ is extensive ("has well-behaved coproducts").
- If F has a nullary operation, i.e., a family of maps

$$c_X : 1 \to FX$$

natural in X (eg, $F = \text{Maybe}$)
or a binary commutative operation, i.e., a family of maps

$$c_X : X \times X \to FX$$

natural in X such that



(eg, $F = \mathcal{M}^{+}_{\text{fin}}$) and F interacts with G, then $GY \cong 0$.

A degeneracy thm for mnd-comnd int laws

- If T has a binary associative operation, ie a family of maps $c_X : X \times X \to TX$ natural in X such that

$$
\begin{array}{ccc}
(X \times X) \times X & \xrightarrow{\ \ell_X\ } & TX \\
{\scriptstyle \text{ass}}\downarrow & & \\
X \times (X \times X) & \xrightarrow{\ r_X\ } & TX
\end{array}
$$

where

$$
\begin{array}{l}
\ell_X = (X \times X) \times X \xrightarrow{c_X \times \eta_X} TX \times TX \xrightarrow{c_{TX}} TTX \xrightarrow{\mu_X} TX \\
r_X = X \times (X \times X) \xrightarrow{\eta_X \times c_X} TX \times TX \xrightarrow{c_{TX}} TTX \xrightarrow{\mu_X} TX
\end{array}
$$

(eg, $T = \mathrm{List}^+$), then any int law ψ of T and D obeys

$$
\begin{array}{ccccc}
(X \times X) \times X \times DY & \xrightarrow{\ell_X \times \mathrm{id}} & TX \times DY & \xrightarrow{\psi_{X,Y}} & X \times Y \\
{\scriptstyle \mathrm{fst} \times \mathrm{id} \times \mathrm{id}}\downarrow & & & & \\
X \times X \times DY & \xrightarrow{c_X \times \mathrm{id}} & TX \times DY & \xrightarrow{\psi_{X,Y}} & X \times Y \\
{\scriptstyle \mathrm{id} \times \mathrm{snd} \times \mathrm{id}}\uparrow & & & & \\
X \times (X \times X) \times DY & \xrightarrow{r_X \times \mathrm{id}} & TX \times DY & \xrightarrow{\psi_{X,Y}} & X \times Y
\end{array}
$$

Residual functor-functor interaction laws

- Given a monad (R, η^R, μ^R) on $\mathcal{C}$.
- Eg, $R = \text{Maybe}$, $\mathcal{M}^+_{\text{fin}}$ or $\mathcal{M}_{\text{fin}}$.

- A *residual functor-functor interaction law* is given by two functors $F, G : \mathcal{C} \to \mathcal{C}$ and a family of maps

$$\phi_{X,Y} : FX \times GY \to R(X \times Y)$$

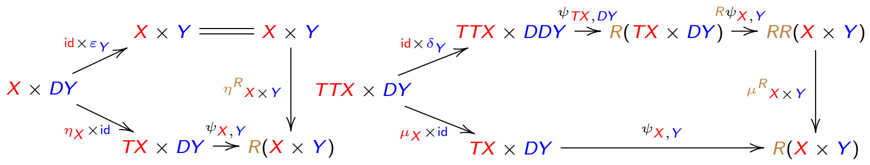
natural in X, Y.

Residual monad-comonad interaction laws

- A *residual monad-comonad interaction law* is given by a monad (T, η, μ), a comonad (D, ε, δ) and a family of maps

$$\psi_{X,Y} : TX \times DY \to R(X \times Y)$$

natural in X, Y such that



- R-residual functor-functor interaction laws form a monoidal category with R-residual monad-comonad interaction laws as monoids.

Intensional nondeterminism incl nullary choice

- $TX = \mu Z.\, X + \mathsf{Maybe}\,(Z \times Z)$
  (wellfounded nullary-binary leaf trees)

  $DY = \nu W.\, Y \times (W + W)$

- $RZ = \mathsf{Maybe}\, Z$

  Both T and R are free monads.

- $\psi_{X,Y} : TX \times DY \to R(X \times Y)$

  $\psi\,(\mathsf{in}\,(\mathsf{inl}\,x), e) = \mathsf{just}\,(x, \mathsf{fst}\,(\mathsf{out}\,e))$
  $\psi\,(\mathsf{in}\,(\mathsf{inr}\,\mathsf{nothing}), e) = \mathsf{nothing}$

  $$\psi\,(\mathsf{in}\,(\mathsf{inr}\,(\mathsf{just}\,(c_0, c_1))), e) = \mathsf{case}\ \mathsf{snd}\,(\mathsf{out}\,e)\ \mathsf{of}\ \begin{cases} \mathsf{inl}\,e' \mapsto \psi\,(c_0, e') \\ \mathsf{inr}\,e' \mapsto \psi\,(c_1, e') \end{cases}$$

Nontermination

- $TX = \mathrm{Delay}\, X = \nu Z.\, X + Z$
  (nonwellfounded nullary leaf trees)
  – delay monad

  $DY = \mu W.\, Y \times \mathrm{Maybe}\, W$
  – nonempty lists and suffixes comonad, for timeouts

- $RZ = \mathrm{Delay}\, Z$

- $\psi_{X,Y} : TX \times DY \to R(X \times Y)$

  $$\psi\,(c, \mathrm{in}\, e) = \mathrm{case\ out}\, c, e \mathrm{\ of} \left\{ \begin{array}{lcl} \mathrm{inl}\, x, (y, \_) & \mapsto & \mathrm{out}^{-1}\,(\mathrm{inl}\,(x, y)) \\ \mathrm{inr}\, c, (y, \mathrm{nothing}) & \mapsto & \mathrm{out}^{-1}\,(\mathrm{inr}\,(\psi\,(c, (y, \mathrm{nothing})))) \\ \mathrm{inr}\, c, (\_, \mathrm{just}\, e) & \mapsto & \psi\,(c, e) \end{array} \right.$$

Summary

- Monads – notions of computation,

  comonads – notions of environments of computation,

  interaction laws – how computations and environments of some notions work together to yield a return value and a final state.

- Interaction laws are monoids (in a suitable category) just as monads.

- Computations of some notions cannot interact like this, but can interact to yield a computation of a simpler notion.
- Residual interaction laws describe such interactions.

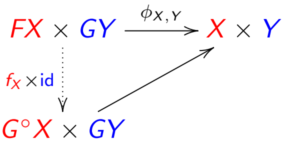
Duals

Duals

- Given a functor/monad/comonad, is there a "greatest" functor/comonad/monad interacting with it?

- Given a functor G, can we find a functor G° and an int law of G°, G so that

  for any functor F and int law ϕ of F, G, there would be a unique nat transf $f : F \to G^\circ$ such that



- The same question makes sense in the presense of a residual monad R.

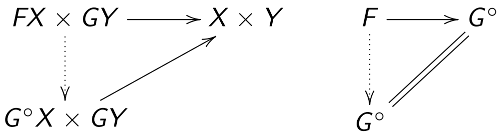
Dual of a functor

- Assume again that $\mathcal{C}$ is Cartesian closed (or symm monoidal closed).
- The "greatest" functor interacting with a functor $G : \mathcal{C} \to \mathcal{C}$ is its *dual*[2] $G^\circ : \mathcal{C} \to \mathcal{C}$ defined by

$$G^\circ X = \int_Y GY \Rightarrow (X \times Y)$$

(if this end exists).

$$\frac{FX \times GY \to X \times Y \text{ nat in } X, Y}{FX \to \underbrace{\int_Y GY \Rightarrow (X \times Y)}_{G^\circ X} \text{ nat in } X}$$



[2] *Co* in Kmett's terminology

Dual of a functor ctd

- $(-)^\circ$ is a functor $[\mathcal{C},\mathcal{C}]^{\mathrm{op}} \to [\mathcal{C},\mathcal{C}]$ (if all functors $\mathcal{C} \to \mathcal{C}$ are dualizable; if not, restrict to some full subcategory of $[\mathcal{C},\mathcal{C}]$ closed under dualization).

- $G^\circ = G \multimap_\star \mathrm{Id}$ where $G \multimap_\star (-)$ is the right adjoint of $(-) \star G$ and $F \star G$ is the Day convolution of F and G.

- These categories are isomorphic:
  - functor-functor interaction laws;
  - pairs of functors F, G with nat. transfs. $F \to G^\circ$;
  - pairs of functors F, G with nat. transfs. $G \to F^\circ$.

Some examples of dual

- If $\mathcal{C}$ is distributive, then
- for $GY = 0$, we have $G^\circ X \cong 1$;
- for $GY = G_0 Y + G_1 Y$, we have $G^\circ X \cong G_0^\circ X \times G_1^\circ X$.

- If $\mathcal{C}$ is Cartesian closed (or symm mon closed) and well-pointed, then
- for $GY = 1$, we have $G^\circ X \cong 0$;
- for $GY = A \times G' Y$, we have $G^\circ X \cong A \Rightarrow G'^\circ X$;

- for $GY = A \Rightarrow Y$, we have $G^\circ X \cong A \times X$;
- for $GY = A \Rightarrow G' Y$, we only have $G^\circ X \leftarrow A \times G'^\circ X$;

- $\mathrm{Id}^\circ \cong \mathrm{Id}$;
- but we only have $(G_0 \cdot G_1)^\circ \leftarrow G_0^\circ \cdot G_1^\circ$.

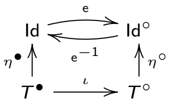
- If $\mathcal{C}$ is extensive, then
  for any G with a nullary or a binary commutative operation, we have
  $G^\circ X \cong 0$.

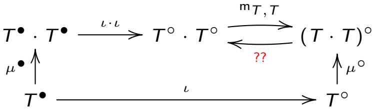
Dual of a comonad / Sweedler dual of a monad

- The dual D° of a comonad D is a monad.
- This is because $(-)^\circ : [\mathcal{C}, \mathcal{C}]^{\mathrm{op}} \to [\mathcal{C}, \mathcal{C}]$ is lax monoidal, so sends monoids to monoids.
- But $(-)^\circ$ is not oplax monoidal, does not send comonoids to comonoids. Therefore,
- the dual T° of a monad T is generally not a comonad.
- However, we can talk about the *Sweedler dual* $T^\bullet$ of T, which is.
- Informally, it is defined as the "greatest" functor D that is "smaller" than the functor T° and carries a comonad structure $\eta^\bullet$, $\mu^\bullet$ agreeing with η°, μ°.

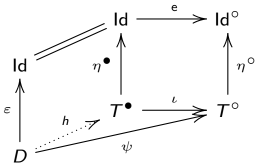
Dual of a comonad / Sweedler dual of a monad ctd

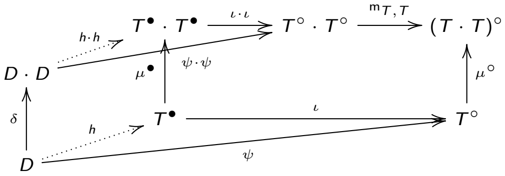
- Formally, the *Sweedler dual* of the monad T is the comonad $(T^\bullet, \eta^\bullet, \mu^\bullet)$ together with a natural transformation $\iota : T^\bullet \to T^\circ$ such that





  and such that, for any comonad (D, ε, δ) together with a natural transformation ψ satisfying the same conditions, there is a unique comonad map $h : D \to T^\bullet$ satisfying





Some examples of dual vs Sweedler dual

- Let $TX = S \Rightarrow (S \times X) \cong (S \Rightarrow S) \times (S \Rightarrow X)$
  – the state monad

- We have $T^\circ Y = (S \Rightarrow S) \Rightarrow (S \times Y)$,
  but $T^\bullet Y = S \times (S \Rightarrow Y)$.

- The func-func int law of T and T° is

  $\phi\,(f, e) = \mathrm{let}\ (s, y) = e\,(\mathrm{fst} \circ f)\ \mathrm{in}\ (\mathrm{snd}\,(f\ s), y)$

  (so the environment first asks what the computation does with the initial data state it wants and only depending on the computation's answer provides one)

- The mnd-comnd int law of T and $T^\bullet$ is what we'd expect:

  $\psi\,(f, (s, g)) = \mathrm{let}\ (s', x) = f\ s\ \mathrm{in}\ (x, g\ s')$

Some examples of dual vs Sweedler dual ctd

- Let $TX = \mathsf{List}^+ X \cong \Sigma n : \mathbb{N}.\,([0..n] \Rightarrow X)$
  – the nonempty lists monad for nondeterminism
- We have $T^\circ Y \cong \Pi n : \mathbb{N}.\,([0..n] \times Y)$,
  but $T^\bullet Y \cong Y \times (Y + Y)$.

- Let $TX = \mu Z.\, X + Z \times Z \cong \Sigma s : S.\,(P\,s \Rightarrow X)$
  ($S \cong T1$ – shapes of binary leaf trees, P – positions in them)
  – the intensional nondeterminism monad
- We have $T^\circ Y \cong \Pi s : S.\,(P\,s \times Y)$,
  but $T^\bullet Y = \nu W.\, Y \times (W + W)$.

- We only have $(F^*)^\circ \leftarrow (F^\circ)^\dagger$, but $(F^*)^\bullet \cong (F^\circ)^\dagger$.

Summary

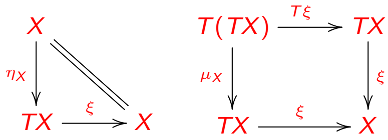
- The greatest functor resp monad interacting with a given functor or comonad is its dual.
- The dual is described by an end formula that simplifies in many specific cases.

- The greatest comonad interacting with a given monad is its Sweedler dual.
- The Sweedler dual is more difficult to find.
- One possibility is to go through the algebraic theory for the monad.

An algebraic-coalgebraic perspective

Monad algebras

- An *algebra* of a monad (T, η, μ) is an object X (*carrier*) with a map $\xi : TX \to X$ (*structure map*) such that



- The algebras of the monad and maps between them form a category **EM**(T), called the *Eilenberg-Moore category*, with an forgetful functor U : **EM**$(T) \to \mathcal{C}$.

- Legend:

  X – a fixed(!) set, values
  $\xi : TX \to X$ – a monomorphic(!) function, crunches a computation into a single value, ie, an effect consumer, a handler

Some examples: Intens'l nondeterminism, nondeterminism

- $TX = \mu Z.X + (Z \times Z)$ – intensional nondeterminism
- $X = M$
  $\xi : TX \to X$
  $\xi\, c = \text{fold}\; op\; c$
  for some M and $op : M \times M \to M$
- $TX = \mu Z.X \times \text{Maybe}\, Z$ – nondeterminism
  (nonempty lists)
- $X = M$
  $\xi : TX \to X$
  $\xi\, c = \text{nelistfold}\; op\; c$
  for some M and $op : M \times M \to M$ forming a *semigroup*
  (*op* needs to be associative)

Comonad coalgebras

- A *coalgebra* of a comonad (D, ε, δ) is an object Y with a map $\chi : Y \to DY$ such that

$$\begin{array}{ccc} Y & & \\ {\scriptstyle\chi}\downarrow & \searrow\!\!\!{=} & \\ DY & \xrightarrow{\varepsilon_Y} & Y \end{array} \qquad \begin{array}{ccc} Y & \xrightarrow{\chi} & DY \\ {\scriptstyle\chi}\downarrow & & \downarrow{\scriptstyle D\chi} \\ DY & \xrightarrow{\delta_Y} & D(DY) \end{array}$$

- The coalgebras of the comonad and maps between them form a category $\mathbf{coEM}(D)$, called the *coEilenberg-Moore category*, with a forgetful functor $U : \mathbf{coEM}(D) \to \mathcal{C}$.

- Legend:

  Y – a fixed(!) set, states
  $\chi : Y \to DY$ – a monomorphic(!) function, blowing a state up into an environment, ie, a coeffect producer, a cohandler

An example: Costate, intensional costate

- $DY = S \times (S \Rightarrow Y)$ – costate for S

  $Y = S$
  $\chi : Y \to DY$
  $\chi\ s = (s, \lambda s'.\, s')$

- $DY = \nu Z.\, Y \times (S \times Y) \times (S \Rightarrow Y)$
  – intensional costate for S

  $Y = Q$
  $\chi : Y \to DY$
  $\chi\ q = \mathrm{out}^{-1}\,(s, \mathrm{let}\ (s, q') = \mathit{coget}\, q\ \mathrm{in}\ (s, \chi\, q'), \lambda s.\, \chi\,(\mathit{coput}\,(q, s)))$
  for some Q and $\mathit{coget} : Q \to S \times Q$, $\mathit{coput} : Q \times S \to Q$ not necessarily forming a lens

Some alternative definitions of int laws

- If $\mathcal{C}$ is Cartesian closed (or symm mon closed), then R-resid mnd-comnd int laws of T, D can be defined in alternative ways:

$$\frac{\dfrac{TX \times DY \to R(X \times Y) \text{ nat in } X, Y \text{ subj to eqs}}{\mathcal{C}(X \times Y, Z) \to \mathcal{C}(TX \times DY, RZ) \text{ nat in } X, Y, Z \text{ subj to eqs}}}{\dfrac{T(Y \Rightarrow Z) \to DY \Rightarrow RZ \text{ nat in } Y, Z \text{ subj to eqs}}{D(X \Rightarrow Z) \to TX \Rightarrow RZ \text{ nat in } X, Z \text{ subj to eqs}}}$$

- Legend:
  X – values
  Y – states
  Z – observables
  (values for residual computations)
  $X \times Y \to Z$ – observation functions

A (co)algebraic view

- Resid mnd-comnd int laws are in a bijection with coalgebra-algebra exponentiation functors:

$$T(Y \Rightarrow Z) \to DY \Rightarrow RZ \text{ nat in } Y, Z \text{ subj to eqs}$$

$$\begin{array}{ccc}
(\mathbf{coEM}(D))^{\mathrm{op}} \times \mathbf{EM}(R) & \longrightarrow & \mathbf{EM}(T) \\
\downarrow {\scriptstyle U^{\mathrm{op}} \times U} & & \downarrow {\scriptstyle U} \\
\mathcal{C}^{\mathrm{op}} \times \mathcal{C} & \xrightarrow{\Rightarrow} & \mathcal{C}
\end{array}$$

$$(Y, \chi : Y \to DY), (Z, \zeta : RZ \to Z) \mapsto (Y \Rightarrow Z, T(Y \Rightarrow Z) \to (Y \Rightarrow Z))$$

$$\begin{array}{ccc}
(\mathbf{coKl}(D))^{\mathrm{op}} \times \mathbf{Kl}(R) & \longrightarrow & \mathbf{EM}(T) \\
\downarrow {\scriptstyle L^{D^{\mathrm{op}}} \times R^{T}} & & \downarrow {\scriptstyle U} \\
\mathcal{C}^{\mathrm{op}} \times \mathcal{C} & \xrightarrow{\Rightarrow} & \mathcal{C}
\end{array}$$

A (co)algebraic view ctd

- Explicitly, given a resid mnd-comnd int law ψ,

  the corresponding (co)alg exp functor E sends
  an EM-coalgebra (Y, χ) of D and an EM-algebra (Z, ζ) of R
  to the EM-algebra $(Y \Rightarrow Z, \xi)$ of T where

$$\xi = \ T(Y \Rightarrow Z) \xrightarrow{\psi_{Y,Z}} DY \Rightarrow RZ \xrightarrow{\chi \Rightarrow \zeta} Y \Rightarrow Z$$

- Conversely, given a (co)alg exp functor E,

  the corresponding resid mnd-comnd int law is

$$\psi_{Y,Z} = \ T(Y \Rightarrow Z) \xrightarrow{T(\varepsilon_Y \Rightarrow \eta^R_Z)} T(DY \Rightarrow RZ) \xrightarrow{e_{Y,Z}} DY \Rightarrow RZ$$

  where $(DY \Rightarrow RZ, e_{Y,Z}) = E((DY, \delta_Y), (RZ, \mu^R_Z))$.

Example of this

- Resid mnd-cmnd int law:

  $TX = \mu Z.X + (Z \times Z) + (S \Rightarrow Z) + (S \times Z)$
  – intensional nondeterminism and state combined

  $DY = S \times (S \Rightarrow Y)$ – costate

  $RX = \mu Z.X + (Z \times Z)$ – intensional nondeterminism

  $\psi_{X,Y} : TX \times DY \to R(X \times Y)$
  $\psi = \ldots$
- Coalgebra (cohandler):

  $Y = S$
  $\chi : Y \to DY$
  $\chi\, s = (s, \lambda s'.\, s')$
- Algebra (residual handler):

  $Z = M$
  $\zeta : RZ \to Z$
  $\zeta\, c = \text{fold}\; op\; c$

  for some M and $op : M \times M \to M$

Intermediate views

- The picture is finer, there are also two intermediate bijections:

$$\mathbf{MCIL}_R(T, D)$$

$$\mathbf{MCIL}_R(T, D) \cong [(\mathbf{coEM}(D))^{\mathrm{op}}, (\mathbf{SRun}_R(T))^{\mathrm{op}}]_{\mathrm{cp.}} \qquad \mathbf{MCIL}_R(T, D) \cong [\mathbf{EM}(R), \mathbf{CRun}_D(T)]_{\mathrm{cp.}}$$

$$[(\mathbf{coEM}(D))^{\mathrm{op}}, (\mathbf{SRun}_R(T))^{\mathrm{op}}]_{\mathrm{cp.}} \cong [(\mathbf{coEM}(D)^{\mathrm{op}} \times \mathbf{EM}(R), \mathbf{EM}(T)]_{\mathrm{ce.}} \cong [\mathbf{EM}(R), \mathbf{CRun}_D(T)]_{\mathrm{cp.}}$$

where

$\mathbf{MCIL}_R(T, D)$ – R-residual mnd-comnd int laws of T, D
$\mathbf{SRun}_R(T)$ – R-residual stateful runners of T
$\mathbf{CRun}_D(T)$ – D-fuelled continuation-based runners of T
cp. – carrier-preserving
ce. – carrier-exponentiating

Stateful runners

- For any Y, we have

R-residual stateful runners of T w/ carrier Y, ie
$TX \times Y \to R(X \times Y)$ nat in X subj to eqs

$$=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=$$

monad morphisms from T to St^R_Y, ie
$TX \to Y \Rightarrow R(X \times Y)$ nat in X subj to eqs

$$=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=\!=$$

$$\begin{array}{ccc} \mathbf{EM}(R) & \longrightarrow & \mathbf{EM}(T) \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathcal{C} & \xrightarrow{Y \Rightarrow (-)} & \mathcal{C} \end{array}$$

where St^R_Y is the *R-transformed state monad*
for state object Y given by

$$\mathrm{St}^R_Y X = Y \Rightarrow R(X \times Y)$$

Continuation-based runners

- For any Z, we have

$$\frac{\dfrac{\begin{array}{c}\textit{D-fuelled continuation-based runners} \text{ of } T \text{ w/ carrier } Z \text{, ie} \\ D(X \Rightarrow Z) \to TX \Rightarrow Z \text{ nat in } X \text{ subj to eqs}\end{array}}{\begin{array}{c}\text{monad morphisms from } T \text{ to } \mathsf{Cnt}^D_Z \text{, ie} \\ TX \to D(X \Rightarrow Z) \Rightarrow Z \text{ nat in } X \text{ subj to eqs}\end{array}}}{\begin{array}{ccc} (\mathbf{coEM}(D))^{\mathrm{op}} & \longrightarrow & \mathbf{EM}(T) \\ {\scriptstyle U}\downarrow & & \downarrow{\scriptstyle U} \\ \mathcal{C} & \xrightarrow{(-)\Rightarrow Z} & \mathcal{C} \end{array}}$$

where Cnt^D_Z is the *D-transformed continuation monad* for answer object Z given by

$$\mathsf{Cnt}^D_Z X = D(X \Rightarrow Z) \Rightarrow Z$$

EM algebras of T w/ carrier $Y \Rightarrow Z$ as runners

- For any Y, Z, we have

$$\frac{\dfrac{\text{state and continuation based runners of } T \text{ w/ carrier } Z\text{, ie } \mathcal{C}(X \times Y, Z) \to \mathcal{C}(TX \times Y, Z) \text{ nat in } X \text{ subj to eqs}}{\text{monad morphisms from } T \text{ to } \mathrm{xCntSt}_{Y,Z} \cong \mathrm{xCostCnt}_{Y,Z}\text{, ie } TX \to Y \Rightarrow \mathrm{xCnt}_Z(X \times Y) \cong \mathrm{xCost}_Y(X \Rightarrow Z) \Rightarrow Z \text{ nat in } X \text{ subj to eqs}}}{\text{EM algebras of } T \text{ with carrier } Y \Rightarrow Z}$$

where

$$\begin{array}{rcl}
\mathrm{xCnt}_Z X & = & \mathcal{C}(X, Z) \pitchfork Z \\
\mathrm{xCntSt}_{Y,Z} X & = & Y \Rightarrow \mathrm{xCnt}_Z(X \times Y) \\
 & = & Y \Rightarrow (\mathcal{C}(X \times Y, Z) \pitchfork Z) \\
\mathrm{xCost}_Y X & = & \mathcal{C}(Y, X) \bullet Y \\
\mathrm{xCostCnt}_{Y,Z} X & = & \mathrm{xCost}_Y(X \Rightarrow Z) \Rightarrow Z \\
 & = & (\mathcal{C}(Y, X \Rightarrow Z) \bullet Y) \Rightarrow Z
\end{array}$$

Summary

- Monad-comonad int laws are closely related to effect handlers and coeffect producers.
- Combining one with a residual effect handler and a coeffect producer gives a handler of original effects for dependencies of observables on states.
- Runners are obtained by combining a monad-comonad int law with just a coeffect producer or a residual effect handler.

Monoid-comonoid interaction laws

Resid func-func int laws as Chu spaces

- The *Day convolution* of functors F, G is

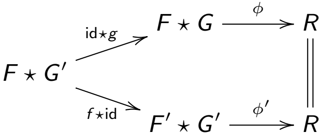
$$(F \star G)Z = \int^{X,Y} \mathcal{C}(X \times Y, Z) \bullet (FX \times GY)$$

(if this coend exists).

- R-resid func-func int laws of F, G are in a bijection with nat tranfs $F \star G \to R$,

$$\begin{array}{c} FX \times GY \to R(X \times Y) \text{ nat in } X, Y \\ \hline\hline \mathcal{C}(X \times Y, Z) \to \mathcal{C}(FX \times GY, RZ) \text{ nat in } X, Y, Z \\ \hline\hline \underbrace{\int^{X,Y} \mathcal{C}(X \times Y, Z) \bullet (FX \times GY)}_{(F \star G)Z} \to RZ \text{ nat in } Z \end{array}$$

- $f : F \to F'$ and $g : G' \to G$ form a R-resid func-func int law map between (F, G, ϕ) and (F', G', ϕ') iff



Resid func-func laws as Chu spaces ctd

- $\star$ is a functor $[\mathcal{C}, \mathcal{C}] \times [\mathcal{C}, \mathcal{C}] \to [\mathcal{C}, \mathcal{C}]$
  (if $\star$ is defined for all functors;
  else restrict $[\mathcal{C}, \mathcal{C}]$ to a suitable full subcategory).
- $(J, \star)$ where $JZ = \mathcal{C}(1, X) \bullet 1$ provide $[\mathcal{C}, \mathcal{C}]$ with a symm monoidal structure.

- These categories are isomorphic for a given monad R:
  - R-residual functor-functor interaction laws;
  - *Chu spaces* on the symm mon category $([\mathcal{C}, \mathcal{C}], J, \star)$ with vertex R,
    ie, pairs of functors F, G with a nat transf $F \star G \to R$
    (with maps between them as described on the previous slide)

Residual mnd-comnd int laws as Chu monoids

- To characterize R-resid mnd-comnd int laws as monoids in $\mathbf{Chu}(R)$, we use that $(\mathsf{Id}, \cdot, J, \star)$ constitute a *duoidal* structure on $[\mathcal{C}, \mathcal{C}]$.
- In particular, $\star$ is oplax monoidal wrt $(\mathsf{Id}, \cdot)$, so there are structural laws

$$\mathsf{Id} \star \mathsf{Id} \to \mathsf{Id}$$
$$(F \cdot F') \star (G \cdot G') \to (F \star G) \cdot (F' \star G')$$

  satisfying equations.
- The duoidal structure on $[\mathcal{C}, \mathcal{C}]$ induces a monoidal structure on $\mathbf{Chu}(R)$ based on $(\mathsf{Id}, \cdot)$.
- The unit is

$$\mathsf{Id} \star \mathsf{Id} \longrightarrow \mathsf{Id} \xrightarrow{\eta^R} R$$

- The tensor of $\phi : F \star G \to R$ and $\phi' : F' \star G' \to R$ is

$$(F \cdot F') \star (G \cdot G') \longrightarrow (F \star G) \cdot (F' \star G') \xrightarrow{\phi \cdot \phi'} R \cdot R \xrightarrow{\mu^R} R$$

- R-residual monad-comonad interaction laws are monoid objects of $\mathbf{Chu}(R)$ wrt this monoidal structure.

Day convolution of two comonads

- As $\star$ is oplax monoidal wrt $(\mathrm{Id}, \cdot)$, it lifts from functors to comonads:

$$\begin{array}{ccc} \mathbf{Comnd}(\mathcal{C}) \times \mathbf{Comnd}(\mathcal{C}) & \xrightarrow{\star} & \mathbf{Comnd}(\mathcal{C}) \\ \Big\downarrow{\scriptstyle U \times U} & & \Big\downarrow{\scriptstyle U} \\ [\mathcal{C}, \mathcal{C}] \times [\mathcal{C}, \mathcal{C}] & \xrightarrow{\star} & [\mathcal{C}, \mathcal{C}] \end{array}$$

- Explicitly, the Day convolution of comonads $(D_0, \varepsilon_0, \delta_0)$ and $(D_1, \varepsilon_1, \delta_1)$ is (D, ε, δ) where $D = D_0 \star D_1$ and

$$\varepsilon = \ D_0 \star D_1 \xrightarrow{\varepsilon_0 \star \varepsilon_1} \mathrm{Id} \star \mathrm{Id} \longrightarrow \mathrm{Id}$$

$$\delta = \ D_0 \star D_1 \xrightarrow{\delta_0 \star \delta_1} (D_0 \cdot D_0) \star (D_1 \cdot D_1) \longrightarrow (D_0 \star D_1) \cdot (D_0 \cdot D_1)$$

- Eg, if $\mathcal{C}$ is well-pointed, the Day convolution of the costate comonads for S_0 and S_1 is the costate comonad for $S_0 \times S_1$.

Cascading

- The Day convolution of functors and comonads allows func-func resp mnd-comnd interactions to be "cascaded".
- Given two mnd-comnd int laws

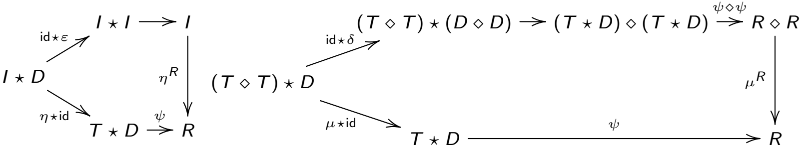
$$\psi : T \star D \to R$$
$$\psi' : R \star D' \to R'$$

one gets a new mnd-comnd int law via

$$T \star (D \star D') \longrightarrow (T \star D) \star D' \xrightarrow{\psi \star D'} R \star D' \xrightarrow{\psi'} R'$$

General residual interaction laws

- Instead of an endofunctor category, one can consider *any* duoidal category $(\mathcal{D}, I, \diamond, J, \star)$.
- Given a monoid object (R, η^R, μ^R) wrt $(I, \diamond)$, we get a $(I, \diamond)$-based monoidal structure on $\mathbf{Chu}(R)$.
- An R-residual monoid-comonoid interaction law is a monoid object of $\mathbf{Chu}(R)$.
- Explicitly, it is given by a monoid (T, η, μ), a comonoid (D, ε, δ) and a map $\psi : T \star D \to R$ such that



Sweedler theory for duoidal categories

- Lopez Franco and Vasilakopoulou generalized Sweedler theory from SMCs to duoidal categories.
- Assume a duoidal category $(\mathcal{D}, I, \diamond, J, \star)$ closed wrt. $(J, \star)$, ie, with a functor $\multimap\!\!\star\colon \mathcal{D}^{\mathrm{op}} \times D \to \mathcal{D}$ such that $- \star G \dashv G \multimap\!\!\star -$.
- For $\mathcal{D} = [\mathcal{C}, \mathcal{C}]$, we have

$$(G \multimap\!\!\star R)X = \int_Y GY \Rightarrow R(X \times Y)$$

- The functors

$$\begin{aligned} \star &: \mathcal{D} \times \mathcal{D} \to \mathcal{D} \\ \multimap\!\!\star &: \mathcal{D}^{\mathrm{op}} \times \mathcal{D} \to \mathcal{D} \end{aligned}$$

  lift to

$$\begin{aligned} \star &: \mathbf{Comon}(\mathcal{D}) \times \mathbf{Comon}(\mathcal{D}) \to \mathbf{Comon}(\mathcal{D}) && \text{tensor of comonoids} \\ \multimap\!\!\star &: (\mathbf{Comon}(\mathcal{D}))^{\mathrm{op}} \times \mathbf{Mon}(\mathcal{D}) \to \mathbf{Mon}(\mathcal{D}) && \textit{power/R-residual dual} \end{aligned}$$

- A R-residual mon-comon int law of T, D is a *measuring map*, ie, a map $UT \star UD \to UR$ such that the transpose $T \to D \multimap\!\!\star R$ is a monoid map.

Sweedler theory for duoidal categories ctd

- If the appropriate adjoints exist, one moreover has functors

$$\begin{array}{rll} \mathrm{H{\small OM}} : (\mathbf{Comon}(\mathcal{D}))^{\mathrm{op}} \times \mathbf{Comon}(\mathcal{D}) \to \mathbf{Comon}(\mathcal{D}) & \text{int hom of comonoids} \\ \triangleright : \mathbf{Comon}(\mathcal{D}) \times \mathbf{Mon}(\mathcal{D}) \to \mathbf{Mon}(\mathcal{D}) & \textit{Sweedler copower} \\ \{-,-\} : (\mathbf{Mon}(\mathcal{D}))^{\mathrm{op}} \times \mathbf{Mon}(\mathcal{D}) \to \mathbf{Comon}(\mathcal{D}) & \textit{Sweedler hom} \end{array}$$

$$\frac{D_0 \star D_1 \to D \text{ in } \mathbf{Comon}(\mathcal{D})}{D_0 \to \mathrm{H{\small OM}}(D_1, D) \text{ in } \mathbf{Comon}(\mathcal{D})} \qquad \begin{array}{c} T \to D \multimap\!\!\star\, R \text{ in } \mathbf{Mon}(\mathcal{D}) \\ \hline\hline UT \star UD \to UR \text{ in } \mathcal{D} \text{ measuring} \\ \hline\hline D \triangleright T \to R \text{ in } \mathbf{Mon}(\mathcal{D}) \\ \hline\hline D \to \{T, R\} \text{ in } \mathbf{Comon}(\mathcal{D}) \end{array}$$

- $D^\circ = D \multimap\!\!\star\, I$ is called the *dual* of D, $D^\bullet = \{T, I\}$ the *Sweedler dual* of T.

- The category $(\mathbf{Comon}(\mathcal{D}), J, \star, \mathrm{H{\small OM}})$ is symmetric monoidal closed.
- The category $(\mathbf{Mon}(\mathcal{D}), \{-,-\}, \triangleright, \multimap\!\!\star)$ is enriched, copowered and powered over $(\mathbf{Comon}(\mathcal{D}), J, \star, \mathrm{H{\small OM}})$.

Mon-comon int laws of free monoids

- Exploiting the Sweedler theory perspective, some things about monoid-comonoid interaction become very easy to calculate.
- Eg, for $T = F^*$ (the free monoid on an object F), the R-residual mon-comon int laws with D are just obj-obj int laws:

$$\begin{array}{c}
F \star UD \to UR \text{ in } \mathcal{D} \\
\hline\hline
F \to UD \mathrel{-\!\!\star} UR \text{ in } \mathcal{D} \\
\hline\hline
F \to U(D \mathrel{-\!\!\star} R) \text{ in } \mathcal{D} \\
\hline\hline
F^* \to D \mathrel{-\!\!\star} R \text{ in } \mathbf{Mon}(\mathcal{D}) \\
\hline\hline
U(F^*) \star UD \to UR \text{ in } \mathcal{D} \text{ measuring}
\end{array}$$

Sweedler hom from a free monoid

- In a similar vein, for $T = F^*$ (the free monoid on an object F), we can find with ease that the Sweedler hom $\{F^*, R\}$ is $(F \multimap UR)^\dagger$ (a cofree comonoid).

  We have, naturally in a comonoid D,

$$\dfrac{D \to (F \multimap UR)^\dagger \text{ in } \mathbf{Comon}(\mathcal{D})}{\dfrac{UD \to F \multimap UR \text{ in } \mathcal{D}}{\dfrac{F \to UD \multimap UR \text{ in } \mathcal{D}}{\dfrac{F \to U(D \multimap R) \text{ in } \mathcal{D}}{\dfrac{F^* \to D \multimap R \text{ in } \mathbf{Mon}(\mathcal{D})}{D \to \{F^*, R\} \text{ in } \mathbf{Comon}(\mathcal{D})}}}}}$$

  Hence $\{F^*, R\} \cong (F \multimap UR)^\dagger$.

Sweedler hom from a general monad

- Some things can be calculated specifically for monad-comonad interaction combining the Sweedler theory perspective with the (co)algebraic one.

- Eg, the (co)algebraic perspective characterizes a certain category of runners as a pullback in **CAT**, but then Sweedler theory says the same pullback is the coEM category of the Sweedler hom $\{T, R\}$ (if this comonad exists).

- Specifically, for any Y,

$$\frac{\dfrac{\text{R-residual stateful runners of T w/ carrier Y, ie} \quad TX \times Y \to R(X \times Y) \text{ nat in X subj to eqs}}{\text{monad morphisms from T to } \mathrm{St}_Y^R\text{, ie} \quad TX \to Y \Rightarrow R(X \times Y) \text{ nat in X subj to eqs}}}{\text{coalgebras of $\{T, R\}$ with carrier Y}}$$

Summary

- Functor-functor and monad-comonad interaction laws generalize to object-object resp monoid-comonoid interaction laws in duoidal categories.

- This kind of interaction laws, in particular "greatest" interacting monoids, comonoids, "smallest" residual monoids have been studied in algebra, in Sweedler theory.
- The Sweedler theory perspective allows working with interaction laws at a very high level.

- For certain calculations specifically for monad-comonad interaction laws, the Sweedler theory perspective can be combined with the (co)algebraic view.

References (1)

- J. Power, O. Shkaravska. From comodels to coalgebras: state and arrays. ENTCS 2004 (CMCS '04).
- G. Plotkin, J. Power. Tensors of comodels and models for operational Semantics. ENTCS 2008 (MFPS '08).
- R. E. Møgelberg, S. Staton. Linear usage of state. LMCS 2014.
- T. Uustalu. Stateful runners of effectful computations. ENTCS 2015. (MFPS '15).
- D. Ahman, A. Bauer. Runners in action. ESOP '20.
- S. Katsumata, E. Rivas, T. Uustalu. Interaction laws of monads and comonads. LICS '20.
- T. Uustalu, N. Voorneveld. Algebraic and coalgebraic perspectives on interaction laws. APLAS '20.
- R. Garner. The costructure-cosemantics adjunction for comodels. arXiv, 2020.

References (2)

- E. Kmett. Monads from comonads. A series of blog posts at comonad.com/reader, 2011.
- D. Piponi. Cofree meets free. Blog post at blog.sigfpe.com, 2014.
- P. Freeman. Comonads as spaces. A series of blog posts at blog.functorial.com, 2016.
- A. Xavier. Comonads for user interfaces. BSc thesis, U Federal de Minas Gerais, 2017/18.