Monads and More: Part 4

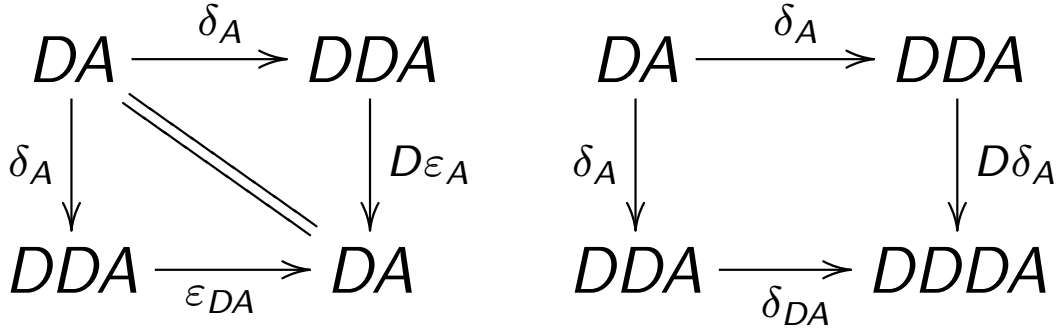
Tarmo Uustalu, Institute of Cybernetics, Tallinn

University of Nottingham, 14–18 May 2007
University of Udine, 2–6 July 2007

Comonads

- Comonads are the dual of monads.
- A *comonad* is a
  - a functor $D : \mathcal{C} \to \mathcal{C}$ (the *underlying functor*),
  - a natural transformation $\eta : D \dot{\to} \mathrm{Id}_{\mathcal{C}}$ (the *counit*),
  - a natural transformation $\delta : D \dot{\to} DD$ (the *comultiplication*)

  satisfying these conditions:



- In other words, a comonad is comonoid in $[\mathcal{C}, \mathcal{C}]$ (a monoid in $[\mathcal{C}, \mathcal{C}]^{\mathrm{op}}$).

CoKleisli triples

- A *coKleisli triple* is given by
  - an object mapping $D : |\mathcal{C}| \to |\mathcal{C}|$,
  - for any object A, a map $\varepsilon_A : DA \to A$,
  - for any map $k : DA \to B$, a map $k^\dagger : DA \to DB$ (the *coKleisli extension* operation)

  satisfying
  - if $k : DA \to B$, then $\varepsilon_B \circ k^\dagger = k$,
  - $\varepsilon_A^\dagger = \mathrm{id}_{DA}$,
  - if $k : DA \to B$, $\ell : DB \to C$, then $(\ell \circ k^\dagger)^\dagger = \ell^\dagger \circ k^\dagger$.
- There is a bijection between comonads and coKleisli triples.

CoKleisli category of a comonad

- A comonad D on a category $\mathcal{C}$ induces a category $\mathbf{CoKl}(D)$ called the *coKleisli category* of D defined by
  - an object is an object of $\mathcal{C}$,
  - a map of from A to B is a map of $\mathcal{C}$ from DA to B,
  - $\mathrm{id}^D_A =_{\mathrm{df}} DA \xrightarrow{\varepsilon_A} A$,
  - if $k : A \to^D B$, $\ell : B \to^D C$, then $\ell \circ^D k =_{\mathrm{df}} DA \xrightarrow{\mu_A} DDA \xrightarrow{Dk} DB \xrightarrow{\ell} C$.
- From $\mathcal{C}$ there is an identity-on-objects *inclusion functor* J to $\mathbf{CoKl}(D)$, defined on maps by
  - if $f : A \to B$, then $Jf =_{\mathrm{df}} DA \xrightarrow{\varepsilon_A} A \xrightarrow{f} B = DA \xrightarrow{Df} DB \xrightarrow{\varepsilon_B} B$.
- The functor J has a left adjoint $U : \mathbf{CoKl}(D) \to \mathcal{C}$ given by $UA =_{\mathrm{df}} DA$, if $k : A \to^D B$, then $Uk =_{\mathrm{df}} DA \xrightarrow{k^\dagger} DB$.

Comonadic notions of computation

- We think of $\mathcal{C}$ as the category of pure functions and of DA as the type of coeffectful computations of values of type A (values in context).
- **CoKl**(D) is the category of coeffectful or context-dependent functions.
- $\varepsilon_A : DA \longrightarrow A$ is the identity on A seen as trivially context-dependent (discarding the context).
- $Jf : DA \longrightarrow B$ is a general pure function $f : A \longrightarrow B$ regarded as trivially context-dependen.
- $\delta_A : DA \longrightarrow DDA$ blows the context of a value up (duplicates the context).
- $k^\dagger : DA \longrightarrow DB$ is a context-dependent function $k : DA \longrightarrow B$ extended into one that can output a value of in a context (e.g., for a postcomposed context-dependent function).

Examples

- Product comonad, for dependency on an environment:
  - $DA =_{\mathrm{df}} A \times E$ where E is an object of $\mathcal{C}$,
  - $\varepsilon_A =_{\mathrm{df}} A \times E \xrightarrow{\mathsf{fst}} A$,
  - $\delta_A =_{\mathrm{df}} A \times E \xrightarrow{\langle \mathsf{id},\mathsf{snd} \rangle} (A \times E) \times E$,
  - if $k : A \times E \to B$, then $k^\dagger =_{\mathrm{df}} A \times E \xrightarrow{\langle k,\mathsf{snd} \rangle} B \times E$.
- This is the dual of the exceptions monad.
- It is not very interesting, as $\mathbf{CoKl}(D) \cong \mathbf{Kl}(T)$ for $TA =_{\mathrm{df}} E \Rightarrow A$ (the reader monad).

- Exponent comonad:
  - $DA =_{\mathrm{df}} E \Rightarrow A$ where (E, e, m) is a monoid in $\mathcal{C}$,
  - $\varepsilon_A =_{\mathrm{df}} (E \Rightarrow A) \xrightarrow{\mathsf{ur}^{-1}} (E \Rightarrow A) \times 1 \xrightarrow{\mathsf{id} \times e} (E \Rightarrow A) \times E \xrightarrow{\mathsf{ev}} A$,
  - $\delta_A =_{\mathrm{df}} \Lambda(\Lambda(((E \Rightarrow A) \times E) \times E \xrightarrow{\mathsf{a}} (E \Rightarrow A) \times (E \times E) \xrightarrow{\mathsf{id} \times m} (E \Rightarrow A) \times E \xrightarrow{\mathsf{ev}} A))$,
- Interesting special cases are $(E, e, m) =_{\mathrm{df}} (\mathsf{Nat}, 0, +)$ and $(E, e, m) =_{\mathrm{df}} (\mathsf{Nat}, 0, \mathsf{max})$.

- "Costate" comonad:
  - $DA =_{\mathrm{df}} (S \Rightarrow A) \times S$ where S is an object of $\mathcal{C}$,
  - $\varepsilon_A =_{\mathrm{df}} (S \Rightarrow A) \times S \xrightarrow{\mathsf{ev}} A$,
  - if $k : (S \Rightarrow A) \times S \to B$, then
    $k^\dagger =_{\mathrm{df}} (S \Rightarrow A) \times S \xrightarrow{\Lambda(k)\times \mathsf{id}} (S \Rightarrow B) \times S$.
- This comonad arises from the adjunction $S \times - \dashv S \Rightarrow -$.

Symmetric monoidal functors

- A *strong/lax symmetric monoidal functor* between symmetric monoidal categories $(\mathcal{C}, I, \otimes)$ and $(^{D}, I', \otimes')$ is
  - a functor on $F : \mathcal{C} \to^{D}$
  - together with an isomorphism/map $\mathrm{e} : I' \to FI$
  - and a natural isomorphism/transformation with components $\mathrm{m}_{A,B} : FA \otimes' FB \to F(A \otimes B)$

  satisfying

$$\begin{array}{ccccc} FA \otimes' I' & \xrightarrow{\mathrm{id}\otimes'\mathrm{e}'} & FA \otimes' FI & \xrightarrow{\mathrm{m}_{A,I}} & F(A \otimes I) \\ \downarrow{\scriptstyle \mathrm{ur}'_{FA}} & & & & \downarrow{\scriptstyle F\mathrm{ur}_A} \\ FA & & = & & FA \end{array} \qquad \begin{array}{ccc} FA \otimes' FB & \xrightarrow{\mathrm{m}_{A,B}} & F(A \otimes B) \\ \downarrow{\scriptstyle \mathrm{c}'_{FA,FB}} & & \downarrow{\scriptstyle F\mathrm{c}_{A,B}} \\ FB \otimes' FA & \xrightarrow[\mathrm{m}_{B,A}]{} & F(B \otimes A) \end{array}$$

$$\begin{array}{ccccc} (FA \otimes' FB) \otimes' FC & \xrightarrow{\mathrm{m}_{A,B}\otimes\mathrm{id}} & F(A \otimes B) \otimes' FC & \xrightarrow{\mathrm{m}_{A\otimes B,C}} & F((A \otimes B) \otimes C) \\ \downarrow{\scriptstyle \mathrm{a}'_{FA,FB,FC}} & & & & \downarrow{\scriptstyle F\mathrm{a}_{A,B,C}} \\ FA \otimes' (FB \otimes' FC) & \xrightarrow[\mathrm{id}\otimes\mathrm{m}_{B,C}]{} & FA \otimes' F(B \otimes C) & \xrightarrow[\mathrm{m}_{A,B\otimes C}]{} & F(A \otimes (B \otimes C)) \end{array}$$

- A *symmetric monoidal natural transformation* between two (strong or lax) symmetric monoidal functors $(F, \mathrm{e}, \mathrm{m})$, $(G, \mathrm{e}', \mathrm{m}')$ is a natural transformation $\tau : F \dot{\to} G$ satisfying

$$\begin{array}{ccc} I' & \xrightarrow{\mathrm{e}} & FI \\ \| & & \downarrow{\scriptstyle \tau_I} \\ I' & \xrightarrow[\mathrm{e}']{} & GI \end{array} \qquad \begin{array}{ccc} FA \otimes' FB & \xrightarrow{\mathrm{m}_{A,B}} & F(A \otimes B) \\ {\scriptstyle \tau_A \otimes' \tau_B}\downarrow & & \downarrow{\scriptstyle \tau_{A \otimes B}} \\ GA \otimes' GB & \xrightarrow[\mathrm{m}'_{A,B}]{} & G(A \otimes B) \end{array}$$

Symmetric monoidal comonads

- A *strong/lax symmetric monoidal comonad* on a symmmetric monoidal category $(\mathcal{C}, I, \otimes)$ is a comonad (D, ε, δ) where D is a strong/lax symmetric monoidal functor (with I, $\otimes$ preserved by e, m) and ε, δ are symmetric monoidal natural transformations, i.e., satisfy

$$\begin{array}{ccc} I & \xrightarrow{\mathsf{e}} & DI \\ \| & & \downarrow{\scriptstyle \varepsilon_I} \\ I & = & I \end{array} \qquad \begin{array}{ccc} I & \xrightarrow{\quad\mathsf{e}\quad} & DI \\ \| & & \downarrow{\scriptstyle \delta_I} \\ I & \xrightarrow[\mathsf{e}]{} DI & \xrightarrow[D\mathsf{e}]{} DDI \end{array}$$

$$\begin{array}{ccc} DA \otimes DB & \xrightarrow{\mathsf{m}_{A,B}} & D(A \otimes B) \\ \downarrow{\scriptstyle \varepsilon_A \otimes \varepsilon_B} & & \downarrow{\scriptstyle \varepsilon_{A \otimes B}} \\ A \otimes B & = & A \otimes B \end{array}$$

- (Note that Id is always symmetric monoidal and F, G being symmetric monoidal imply that GF is symmetric monoidal too.)
- A strong/lax symmetric *semi*monoidal comonad is as a strong/lax symmetric monoidal comonad, but without e (on a category which may be without I).

Dataflow computations

Dataflow computation = discrete-time signal transformations = stream functions.

The output value at a time instant (stream position) is determined by the input value at the same instant (position) plus further input values.

Example dataflow programs

$$
\begin{aligned}
pos &= 0 \text{ fby } (pos + 1) \\
sum\ x &= x + (0 \text{ fby } (sum\ x)) \\
fact &= 1 \text{ fby } (fact * (pos + 1)) \\
fibo &= 0 \text{ fby } (fibo + (1 \text{ fby } fibo))
\end{aligned}
$$

`pos`	0	1	2	3	4	5	6	...
`sum pos`	0	1	3	6	10	15	21	...
`fact`	1	1	2	6	24	120	720	...
`fibo`	0	1	1	2	3	5	8	...

We want to consider functions Str $A \to$ Str B as impure functions from A to B.

Streams are naturally isomorphic to functions from natural numbers: $\mathrm{Str}A =_{\mathrm{df}} \nu X.A \times X \cong \mathrm{Nat} \Rightarrow A$.

General stream functions $\mathrm{Str}A \to \mathrm{Str}B$ are thus in natural bijection with maps $\mathrm{Str}A \times \mathrm{Nat} \to B$.

Comonad for general stream functions

- Functor:

$$DA =_{\mathrm{df}} (\mathsf{Nat} \Rightarrow A) \times \mathsf{Nat} \cong \mathsf{List}A \times \mathsf{Str}A$$

- Input streams with past/present/future:

$$a_0, a_1, \ldots, a_{n-1}, \boxed{a_n}, a_{n+1}, a_{n+2}, \ldots$$

- Counit:

$$\begin{array}{rcccc} \varepsilon_A & : & (\mathsf{Nat} \Rightarrow A) \times \mathsf{Nat} & \longrightarrow & A \\ & & (a, n) & \mapsto & a(n) \end{array}$$

- CoKleisli extension:

$$\frac{k \ : \ (\mathsf{Nat} \Rightarrow A) \times \mathsf{Nat} \ \longrightarrow \ B}{\begin{array}{rcccc} k^\star & : & (\mathsf{Nat} \Rightarrow A) \times \mathsf{Nat} & \longrightarrow & (\mathsf{Nat} \Rightarrow B) \times \mathsf{Nat} \\ & & (a, n) & \mapsto & (\lambda m\, k(a, m), n) \end{array}}$$

Comonad for causal stream functions

- Functor: $DA =_{\mathrm{df}} \mathrm{NEList} \cong \mathrm{List}A \times A$
- Input streams with past and present but no future
- Counit:

$$\begin{array}{rcccc} \varepsilon_A & : & \mathrm{NEList}A & \longrightarrow & A \\ & & [a_0, \ldots, a_n] & \mapsto & a_n \end{array}$$

- CoKleisli extension:

$$\frac{k \ : \ \mathrm{NEList}A \ \longrightarrow \ B}{\begin{array}{rcccc} k^\star & : & \mathrm{NEList}A & \longrightarrow & \mathrm{NEList}B \\ & & [a_0, \ldots, a_n] & \mapsto & [k[a_0], k[a_0, a_1], \ldots, k[a_0, \ldots, a_n]] \end{array}}$$

Comonad for anticausal stream functions

- Input streams with present and future but no past
- Functor: $DA =_{\mathrm{df}} \mathrm{Str}A \cong A \times \mathrm{Str}A$

Relabelling tree transformations

- Let $H : \mathcal{C} \to \mathcal{C}$. Define $\mathsf{Tree}A =_{\mathrm{df}} \mu X.A \times HX$. We are interested in relabelling functions $\mathsf{Tree}A \to \mathsf{Tree}B$. (Alt. we can define $\mathsf{Tree}^{\infty}A =_{\mathrm{df}} \nu X.A \times HX$ and interest ourselves in relabelling functions $\mathsf{Tree}^{\infty}A \to \mathsf{Tree}^{\infty}B$.)
- Comonad for general relabelling functions:

$$DA =_{\mathrm{df}} \mathsf{Tree}'A \times A \cong \mathsf{Path}A \times \mathsf{Tree}A$$

  where $\mathsf{Path}A =_{\mathrm{df}} \mu X.1 + A \times H'(\mathsf{Tree}A) \times X$ (Huet's zipper).
- E.g., for $HX =_{\mathrm{df}} 1 + X \times X$, $H'X \cong 2 \times X$ and $\mathsf{Path}A \cong \mu X.1 + A \times 2 \times \mathsf{Tree}A \times X$.
- Comonad for bottom-up relabelling functions:

$$DA =_{\mathrm{df}} \mathsf{Tree}A$$

Cartesian preclosed structure of the coKleisli category of a strong/lax (semi)monoidal comonad

- Let D be a comonad on a Cartesian closed category $\mathcal{C}$.
- Since $J : \mathcal{C} \longrightarrow \mathbf{CoKl}(D)$ is a right adjoint and preserves limits, $\mathbf{CoKl}(D)$ inherits products from $\mathcal{C}$. Explicitly, we can define

$$\begin{aligned} A \times^D B &=_{\mathrm{df}} A \times B \\ \pi_0^D &=_{\mathrm{df}} \mathsf{fst} \circ \varepsilon \\ \pi_1^D &=_{\mathrm{df}} \mathsf{snd} \circ \varepsilon \\ \langle k_0, k_1 \rangle^D &=_{\mathrm{df}} \langle k_0, k_1 \rangle \end{aligned}$$

- If D is $(1, \times)$ strong/lax symmetric semimonoidal, then we can also define

$$\begin{array}{rcl} A \Rightarrow^D B & =_{\mathrm{df}} & DA \Rightarrow B \\ \mathsf{ev}^D & =_{\mathrm{df}} & \mathsf{ev} \circ \langle \varepsilon \circ D\mathsf{fst}, D\mathsf{snd} \rangle \\ \Lambda^D(k) & =_{\mathrm{df}} & \Lambda(k \circ \mathsf{m}) \end{array}$$

$$D((DA \Rightarrow B) \times A) \xrightarrow{\langle \varepsilon \circ D\mathsf{fst}, D\mathsf{snd} \rangle} (DA \Rightarrow B) \times DA \xrightarrow{\mathsf{ev}} B$$

$$\frac{DC \times DA \xrightarrow{\mathsf{m}} D(C \times A) \xrightarrow{k} B}{DC \xrightarrow{\Lambda(k \circ \mathsf{m})} DA \Rightarrow B}$$

- Using a strength (if available) is not a good idea: We have no multiplication

$$DC \times DA \xrightarrow{\mathsf{sl}} D(C \times DA) \xrightarrow{D\mathsf{sr}} DD(C \times A) \xrightarrow{?} D(C \times A)$$

and applying ε or $D\varepsilon$ gives a solution where the order of arguments of a function is important and coeffects do not combine:

$$DC \times DA \xrightarrow{\mathsf{id}\times\varepsilon} DC \times A \xrightarrow{\mathsf{sl}} D(C \times A)$$

or

$$DC \times DA \xrightarrow{\varepsilon\times\mathsf{id}} C \times DA \xrightarrow{\mathsf{sr}} D(C \times A)$$

- If D is strong semimonoidal (in which case it is automatically strong symmetric semimonoidal as well), then $A \Rightarrow^D -$ is right adjoint to $- \times^D A$ and hence $\Rightarrow^D$ is an exponent functor:

$$\frac{\frac{D(C \times A) \to B}{DC \times DA \to B}}{DC \to DA \Rightarrow B}$$

- This is the case, e.g., if $DA \cong \nu X.A \times (E \Rightarrow X)$ for some E (e.g., $DA \cong \mathrm{Str}A \cong \nu X.A \times (1 \Rightarrow X)$).

- More typically, D is only lax symmetric semimonoidal.
- Then it suffices to have m satisfying

$$\begin{array}{ccc} DA & = & DA \\ {\scriptstyle \Delta_{DA}}\downarrow & & \downarrow{\scriptstyle D\Delta_A} \\ DA \times DA & \xrightarrow[\mathrm{m}_{A,A}]{} & D(A,A) \end{array}$$

where $\Delta = \langle \mathrm{id}, \mathrm{id} \rangle : A \to A \times A$ is part of the comonoid structure on the objects of $\mathcal{C}$, to get that $m \circ \langle D\mathrm{fst}, D\mathrm{snd} \rangle = \mathrm{id}$ and that $\Rightarrow^D$ is a weak exponent operation on objects. It is not functorial (not even in each argument separately).

Partial uniform parameterized fixpoint operator

Let $F : \mathcal{C} \to \mathcal{C}$. Define $DA =_{\mathrm{df}} \nu Z.A \times FZ$.

Call a coKleisli map $k : A \times B \to^D B$ *guarded* if for some k' we have

$$\begin{array}{ccc} D(A \times B) & \xrightarrow{\quad k \quad} & B \\ \downarrow \cong & & \uparrow k' \\ (A \times B) \times FD(A \times B) & \xrightarrow{\mathsf{fst} \times \mathsf{id}} & A \times FD(A \times B) \end{array}$$

For any guarded $k : A \times B \to^D B$, there is a unique map $\mathrm{fix}(k) : A \to^D B$ satisfying

$$\begin{array}{ccc} A & \xrightarrow{\quad \mathrm{fix}(k) \quad} & B \\ & {\scriptstyle \langle \mathsf{id}^D, \mathrm{fix}(k) \rangle^D} \searrow \quad \nearrow {\scriptstyle k} & \\ & A \times B & \end{array}$$

fix is a partial *Conway operator* defined on guarded maps, i.e., besides the *fixpoint property*, for any guarded $k : A \times^D B \to^D B$,

$$\mathrm{fix}(k) = k \circ^D \langle \mathrm{id}^D, \mathrm{fix}(k) \rangle^D$$

it satisfies *naturality* in A, *dinaturality* in B, and the *diagonal property*: for any guarded $k : A \times^D B \times^D B \to^D B$,

$$\mathrm{fix}(k \circ^D (\mathrm{id}^D \times^D \Delta^D)) = \mathrm{fix}(\mathrm{fix}(k))$$

Wrt. pure maps, fix is also *uniform* (i.e., strongly dinatural in B instead of dinatural), i.e., for any guarded $k : A \times^D B \to^D B$, $\ell : A \times^D B' \to^D B'$ and $h : B \to B'$

$$Jh \circ^D k = \ell \circ^D (\mathrm{id}^D \times^D Jh) \quad \Longrightarrow \quad Jh \circ^D \mathrm{fix}(k) = \mathrm{fix}(\ell)$$

Comonadic semantics

- As in the case of monadic semantics, we interpret the lambda-calculus into **CoKl**(D) in the standard way (using its Cartesian preclosed structure), getting

$$\begin{aligned}
[\![K]\!]^D &=_{\mathrm{df}} \text{an object of } \mathbf{CoKl}(D) \\
&\qquad = \text{that object of } \mathcal{C} \\
[\![A \times B]\!]^D &=_{\mathrm{df}} [\![A]\!]^D \times^D [\![B]\!]^D \\
&\qquad = [\![A]\!]^D \times [\![B]\!]^D \\
[\![A \Rightarrow B]\!]^D &=_{\mathrm{df}} [\![A]\!]^D \Rightarrow^D [\![B]\!]^D \\
&\qquad = D[\![A]\!]^D \Rightarrow [\![B]\!]^D \\
[\![\underline{C}]\!]^D &=_{\mathrm{df}} [\![C_0]\!]^D \times \ldots \times [\![C_{n-1}]\!]^D \\
&\qquad = [\![C_0]\!] \times \ldots \times [\![C_{n-1}]\!]
\end{aligned}$$

$$
\begin{aligned}
[\![(\underline{x})\, x_i]\!]^D &=_{\mathrm{df}} \pi_i^D \\
&= \pi_i \circ \varepsilon \\
[\![(\underline{x})\, \mathit{fst}(t)]\!]^D &=_{\mathrm{df}} \pi_0^D \circ^D [\![(\underline{x})\, t]\!]^D \\
&= \mathsf{fst} \circ [\![(\underline{x})\, t]\!]^D \\
[\![(\underline{x})\, \mathit{snd}(t)]\!]^D &=_{\mathrm{df}} \pi_1^D \circ^D [\![(\underline{x})\, t]\!]^D \\
&= \mathsf{snd} \circ [\![(\underline{x})\, t]\!]^D \\
[\![(\underline{x})\, (t_0, t_1)]\!]^D &=_{\mathrm{df}} \langle [\![(\underline{x})\, t_0]\!]^D, [\![(\underline{x})\, t_1]\!]^D \rangle^D \\
&= \langle [\![(\underline{x})\, t_0]\!]^D, [\![(\underline{x})\, t_1]\!]^D \rangle \\
[\![(\underline{x})\, \lambda x t]\!]^D &=_{\mathrm{df}} \Lambda^D([\![(\underline{x}, x)\, t]\!]^D) \\
&= \Lambda([\![(\underline{x}, x)\, t]\!]^D \circ \mathsf{m}) \\
[\![(\underline{x})\, t\ u]\!]^D &=_{\mathrm{df}} \mathsf{ev}^D \circ^D \langle [\![(\underline{x})\, t]\!]^D, [\![(\underline{x})\, u]\!]^D \rangle^D \\
&= \mathsf{ev} \circ \langle [\![(\underline{x})\, t]\!]^D, ([\![(\underline{x})\, u]\!]^D)^\dagger \rangle
\end{aligned}
$$

- Coeffect-specific constructs are interpreted specifically.
- E.g., for the constructs of a general/causal/anticausal dataflow language we can use the appropriate comonad and define:

$$
\begin{array}{rcl}
[\![(\underline{x})\ t\ \mathit{fby}\ u]\!]^D & =_{\mathrm{df}} & \mathsf{fby} \circ \langle [\![(\underline{x})\ t]\!]^D, ([\![(\underline{x})\ u)]\!]^D)^\dagger \rangle^D \\
[\![(\underline{x})\ t\ \mathit{next}\ u]\!]^D & =_{\mathrm{df}} & \mathsf{next} \circ ([\![(\underline{x})\ t]\!]^D)^\dagger
\end{array}
$$

- Again, we have soundness of typing, in the form $\underline{x} : \underline{C} \vdash t : A$ implies $[\![(\underline{x})t]\!]^D : [\![\underline{C}]\!]^D \to^D [\![A]\!]^D$, but not all equations of the lambda-calculus are validated.
- For a closed term $\vdash t : A$, soundness of typing says that $[\![t]\!]^D : 1 \to^D [\![A]\!]^D$, i.e., $D1 \to [\![A]\!]^D$, so closed terms are evaluated relative to a coeffect over 1.
- In case of general or causal stream functions, an element of $D1$ is a list over 1, i.e., a natural number, the time elapsed.
- If D is strong or lax symmetric monoidal (not just semimonoidal), we have a canonical choice $\mathrm{e} : 1 \to D1$.

- Comonadic dataflow language semantics: The first-order language agrees perfectly with Lucid and Lustre by its semantics.
  The meaning of higher-order dataflow computation has been unclear. We get a neat semantics from mathematical considerations (cf. Colaço, Pouzet's design with two flavors of function spaces).

Symmetric monoidal comonads (and strong monads) in linear / modal logic

- Strong symmetric monoidal comonads are central in the semantics of intuitionistic linear logic and modal logic to interpret the ! and $\Box$ ($\Diamond$) operators.
- Linear logic: Benton, Bierman, de Paiva, Hyland; Bierman; Benton; Mellies; Maneggia; etc.
- Modal logic: Bierman, de Paiva.
- Applications to staged computation and semantics of names: Pfenning, Davies, Nanevski.