Directed Containers

Danel Ahman

(based on joint work with James Chapman and Tarmo Uustalu)

Ljubljana, 11 October 2018

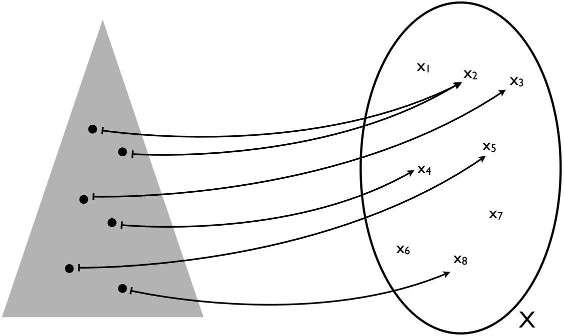
Today's plan

- Directed containers
  - type-theoretic and polynomial presentations
  - their use in functional programming
  - why are they canonical such structure?
- Some constructions on directed containers (see more in papers)
  - coproducts of directed containers
  - strict directed containers and their products
  - focussing a container
- Directed containers and computational effects
- Directed containers and BX
- Directed containers and categories

Prelude

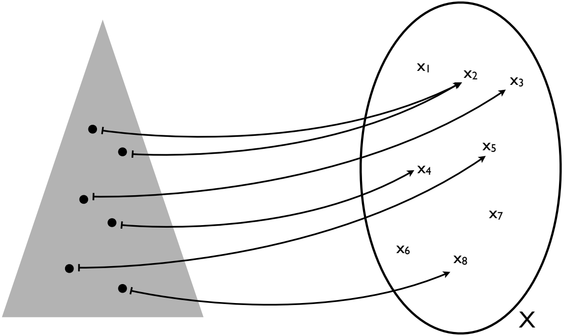
Container syntax of datatypes

- Many datatypes can be represented in terms of
  - shapes and
  - positions in shapes



Container syntax of datatypes

- Many datatypes can be represented in terms of
  - shapes and
  - positions in shapes

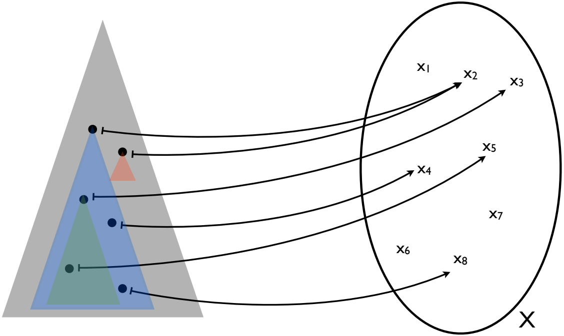


- **Examples:** lists, streams, colists, trees, zippers, etc.

- Containers provide us with a handy syntax to analyse them

Directing containers?

- Containers often exhibit a natural notion of subshape



- Natural questions arise:
  - What is the appropriate specialisation of containers?
  - Does this admit a nice categorical theory?
  - What else is this structure useful for?

Directed containers

Directed containers

- A directed container is given by
  - S : **Set** *(shapes)*
  - $P : S \to$ **Set** *(positions)*

  and

  - $\downarrow\, : \Pi s : S.\, P\, s \to S$ *(subshape)*
  - $\mathrm{o}\, : \Pi\{s : S\}.\, P\, s$ *(root position)*
  - $\oplus : \Pi\{s : S\}.\, \Pi p : P\, s.\, P\, (s \downarrow p) \to P\, s$ *(subshape positions)*

  such that

  - $s \downarrow \mathrm{o} = s$
  - $s \downarrow (p \oplus p') = (s \downarrow p) \downarrow p'$
  - $p \oplus\{s\}\, \mathrm{o} = p$
  - $\mathrm{o}\{s\} \oplus p = p$
  - $(p \oplus\{s\}\, p') \oplus p'' = p \oplus (p' \oplus p'')$

Directed containers

- A directed container is given by
  - $S : \mathbf{Set}$ *(shapes)*
  - $P : S \to \mathbf{Set}$ *(positions)*

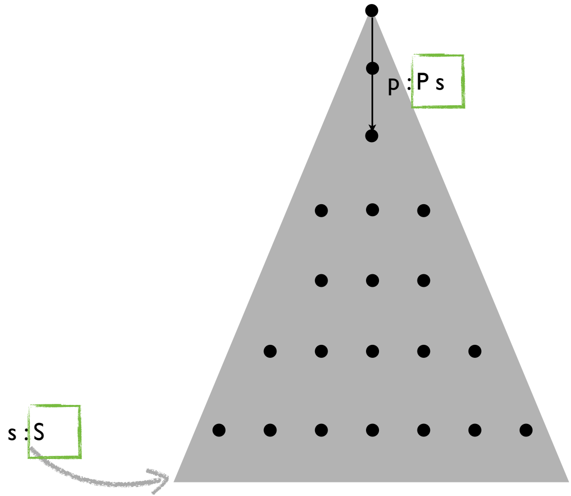
  and

  - $\downarrow : \Pi s : S.\, P\, s \to S$ *(subshape)*
  - $\mathrm{o} : \Pi\{s : S\}.\, P\, s$ *(root position)*
  - $\oplus : \Pi\{s : S\}.\, \Pi p : P\, s.\, P\,(s \downarrow p) \to P\, s$ *(subshape positions)*

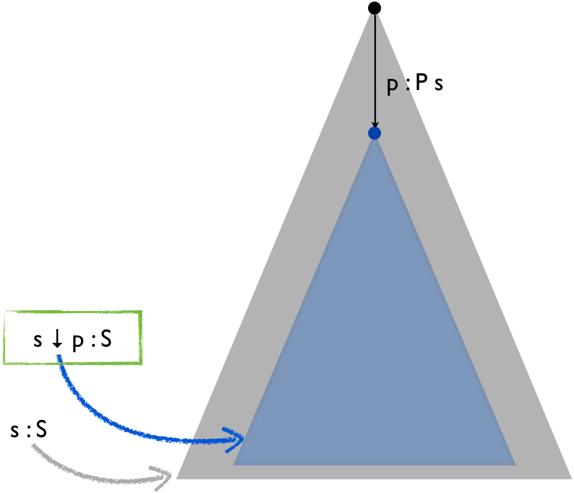
  such that

  - $s \downarrow \mathrm{o} = s$
  - $s \downarrow (p \oplus p') = (s \downarrow p) \downarrow p'$
  - $p \oplus_{\{s\}} \mathrm{o} = p$
  - $\mathrm{o}_{\{s\}} \oplus p = p$
  - $(p \oplus_{\{s\}} p') \oplus p'' = p \oplus (p' \oplus p'')$

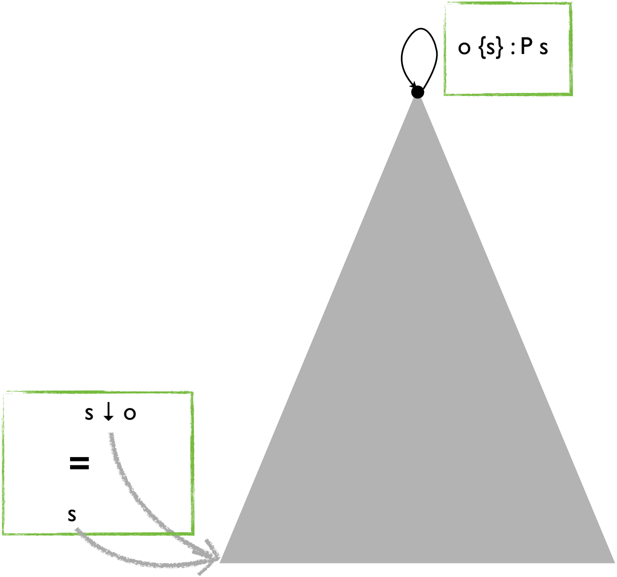
Directed containers illustrated



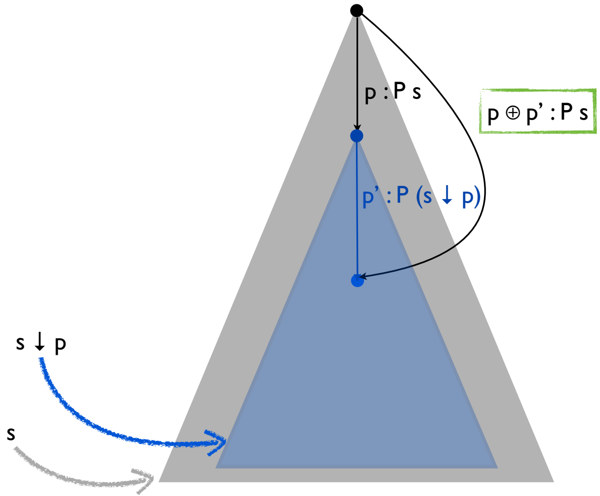
Directed containers illustrated



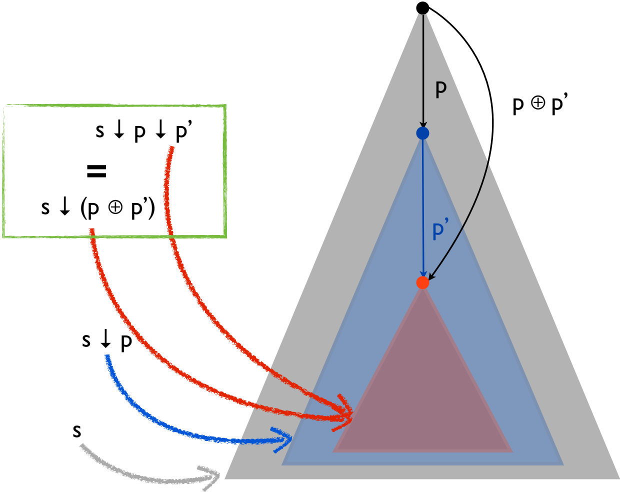
Directed containers illustrated



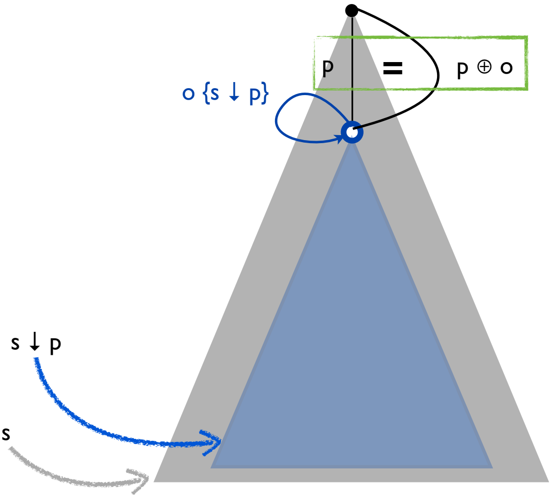
Directed containers illustrated



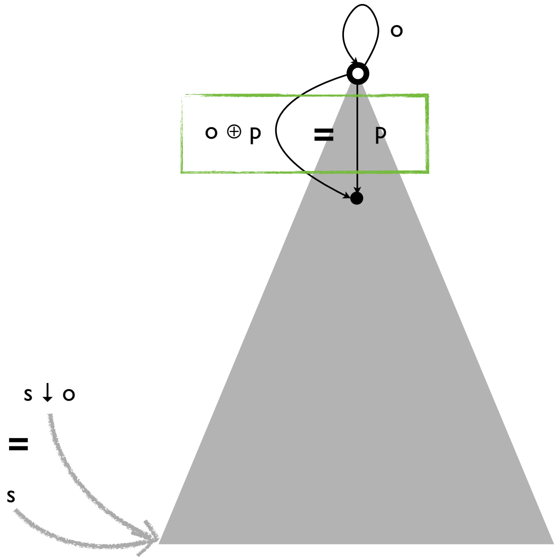
Directed containers illustrated



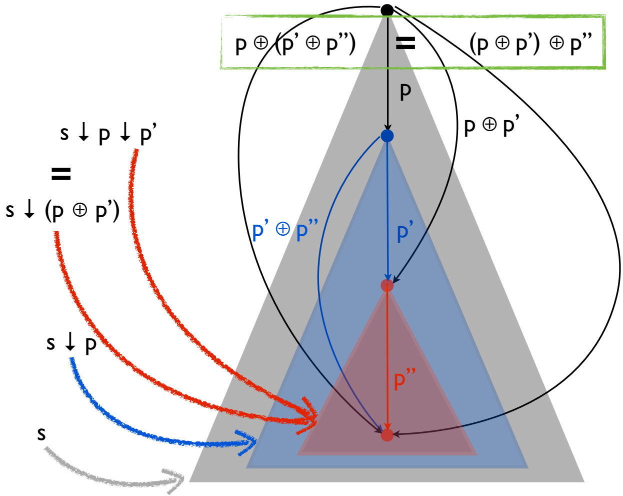
Directed containers illustrated



Directed containers illustrated



Directed containers illustrated



Directed containers (recap)

- A directed container is given by
  - S : **Set** *(shapes)*
  - $P : S \to$ **Set** *(positions)*

  and

  - $\downarrow\, : \Pi s : S.\, P\, s \to S$ *(subshape)*
  - $\mathsf{o} : \Pi\{s : S\}.\, P\, s$ *(root position)*
  - $\oplus : \Pi\{s : S\}.\, \Pi p : P\, s.\, P\,(s \downarrow p) \to P\, s$ *(subshape positions)*

  such that

  - $s \downarrow \mathsf{o} = s$
  - $s \downarrow (p \oplus p') = (s \downarrow p) \downarrow p'$
  - $p \oplus_{\{s\}} \mathsf{o} = p$
  - $\mathsf{o}_{\{s\}} \oplus p = p$
  - $(p \oplus_{\{s\}} p') \oplus p'' = p \oplus (p' \oplus p'')$

Examples: non-empty lists and streams

- Non-empty lists are represented as
  - $S \stackrel{\text{def}}{=} \mathsf{Nat}$ *(shapes)*
  - $P\, s \stackrel{\text{def}}{=} [0..s]$ *(positions)*
  - $s \downarrow p \stackrel{\text{def}}{=} s - p$ *(subshapes)*
  - $\mathsf{o}_{\{s\}} \stackrel{\text{def}}{=} 0$ *(root position)*
  - $p \oplus_{\{s\}} p' \stackrel{\text{def}}{=} p + p'$ *(subshape positions)*

Examples: non-empty lists and streams

- Non-empty lists are represented as
  - $S \stackrel{\text{def}}{=} \mathsf{Nat}$ *(shapes)*
  - $P\,s \stackrel{\text{def}}{=} [0..s]$ *(positions)*
  - $s \downarrow p \stackrel{\text{def}}{=} s - p$ *(subshapes)*
  - $\mathsf{o}_{\{s\}} \stackrel{\text{def}}{=} 0$ *(root position)*
  - $p \oplus_{\{s\}} p' \stackrel{\text{def}}{=} p + p'$ *(subshape positions)*
- Another example is non-empty lists with cyclic shifts

Examples: non-empty lists and streams

- Non-empty lists are represented as
  - $S \stackrel{\text{def}}{=} \mathsf{Nat}$ *(shapes)*
  - $P\,s \stackrel{\text{def}}{=} [0..s]$ *(positions)*
  - $s \downarrow p \stackrel{\text{def}}{=} s - p$ *(subshapes)*
  - $\mathsf{o}_{\{s\}} \stackrel{\text{def}}{=} 0$ *(root position)*
  - $p \oplus_{\{s\}} p' \stackrel{\text{def}}{=} p + p'$ *(subshape positions)*
- Another example is non-empty lists with cyclic shifts
- Streams are represented similarly
  - $S \stackrel{\text{def}}{=} 1$ *(shapes)*
  - $P * \stackrel{\text{def}}{=} \mathsf{Nat}$ *(positions)*
  - ...

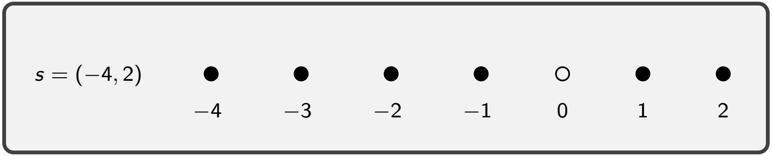
Examples: non-empty lists with a focus

- Zippers – tree-like data-structures consisting of
  - a context and a focal subtree

Examples: non-empty lists with a focus

- Zippers – tree-like data-structures consisting of
  - a context and a focal subtree

- Non-empty lists with a focus
  - $S \stackrel{\text{def}}{=} \text{Nat} \times \text{Nat}$ *(shapes)*
  - $P\,(s_0, s_1) \stackrel{\text{def}}{=} [-s_0..s_1] = [-s_0..-1] \cup [0..s_1]$ *(positions)*



  - $(s_0, s_1) \downarrow p \stackrel{\text{def}}{=} (s_0 + p, s_1 - p)$ *(subshapes)*
  - $o_{\{s_0, s_1\}} \stackrel{\text{def}}{=} 0$ *(root)*
  - $p \oplus_{\{s_0, s_1\}} p' \stackrel{\text{def}}{=} p + p'$ *(subshape positions)*

Directed polynomials

Directed polynomials

- A polynomial (in one variable) is given by

$$1 \xleftarrow{\;!\;} \overline{P} \xrightarrow{\;s\;} S \xrightarrow{\;!\;} 1$$

  where

  - S : **Set** *(shapes)*
  - $\overline{P}$: **Set** *(total positions)*

- Polynomials correspond to containers via $\overline{P} \cong \Sigma\, s : S.\, P\, s$

Directed polynomials

- A polynomial (in one variable) is given by

$$1 \xleftarrow{\;!\;} \overline{P} \xrightarrow{\;\mathsf{s}\;} S \xrightarrow{\;!\;} 1$$

  where

  - S : **Set** *(shapes)*
  - $\overline{P}$: **Set** *(total positions)*

- Polynomials correspond to containers via $\overline{P} \cong \Sigma\, s\!:\!S.\, P\, s$

- A directed polynomial is given by
  - $\mathsf{s} : \overline{P} \longrightarrow S$ (a polynomial)
  - $\downarrow : \overline{P} \longrightarrow S$
  - $\mathsf{o} : S \longrightarrow \overline{P}$ s.t. $\mathsf{s} \circ \mathsf{o} = \mathrm{id}_S$ and $\downarrow \circ\, \mathsf{o} = \mathrm{id}_S$
  - ...

Directed polynomials

- A polynomial (in one variable) is given by

$$1 \xleftarrow{\;!\;} \overline{P} \xrightarrow{\;\mathsf{s}\;} S \xrightarrow{\;!\;} 1$$

  where

  - $S : \mathbf{Set}$ *(shapes)*
  - $\overline{P} : \mathbf{Set}$ *(total positions)*

- Polynomials correspond to containers via $\overline{P} \cong \Sigma\, s : S.\, P\, s$

- A directed polynomial is given by
  - $\mathsf{s} : \overline{P} \longrightarrow S$ (a polynomial)
  - $\downarrow : \overline{P} \longrightarrow S$
  - $\mathsf{o} : S \longrightarrow \overline{P}$ s.t. $\mathsf{s} \circ \mathsf{o} = \mathrm{id}_S$ and $\downarrow \circ\, \mathsf{o} = \mathrm{id}_S$
  - $\ldots$

  - def. is remarkably symmetric in s and $\downarrow$

Directed container morphisms

- A directed container morphism

$$t \lhd q : (S \lhd P, \downarrow, \mathrm{o}, \oplus) \longrightarrow (S' \lhd P', \downarrow', \mathrm{o}', \oplus')$$

  is given by

  - $t : S \to S'$
  - $q : \Pi\{s : S\}.\, P'\,(t\,s) \to P\,s$ (note the direction!)

  such that

  - $t\,(s \downarrow q\,p) = t\,s \downarrow' p$
  - $\mathrm{o}_{\{s\}} = q\,(\mathrm{o}'_{\{t\,s\}})$
  - $q\,p \oplus_{\{s\}} q\,p' = q\,(p \oplus'_{\{t\,s\}} p')$

- Identities and composition are defined component-wise
- Directed containers form a category **DCont**

Directed container morphisms

- A directed container morphism

$$t \lhd q : (S \lhd P, \downarrow, \mathsf{o}, \oplus) \longrightarrow (S' \lhd P', \downarrow', \mathsf{o}', \oplus')$$

  is given by

  - $t : S \to S'$
  - $q : \Pi\{s : S\}.\, P'\,(t\,s) \to P\,s$

  such that

  - $t\,(s \downarrow q\,p) = t\,s \downarrow' p$
  - $\mathsf{o}_{\{s\}} = q\,(\mathsf{o}'_{\{t\,s\}})$
  - $q\,p \oplus_{\{s\}} q\,p' = q\,(p \oplus'_{\{t\,s\}} p')$

- Identities and composition are defined component-wise

- Directed containers form a category **DCont**

Directed containers

=

containers $\cap$ comonads

Interpretation of directed containers

- Any directed container

  $$(S \lhd P, \downarrow, o, \oplus)$$

  defines a functor/~~comonad~~

  $$[\![S \lhd P, \downarrow, o, \oplus]\!]^{\mathrm{dc}} \stackrel{\mathrm{def}}{=} (D, \varepsilon, \delta)$$

  where

  - $D : \mathbf{Set} \longrightarrow \mathbf{Set}$

    $D\,X \stackrel{\mathrm{def}}{=} \Sigma s : S.\,(P\,s \to X)$

  - $\varepsilon_X : D\,X \longrightarrow X$

    $\varepsilon_X\,(s, v) \stackrel{\mathrm{def}}{=} v\,(o_{\{s\}})$

  - $\delta_X : D\,X \longrightarrow D\,D\,X$

    $\delta_X\,(s, v) \stackrel{\mathrm{def}}{=} (s, \lambda p.\,(s \downarrow p, \lambda p'.\,v\,(p \oplus_{\{s\}} p')))$

Interpretation of directed containers

- Any directed container

$$(S \lhd P, \downarrow, \mathsf{o}, \oplus)$$

defines a ~~functor~~ comonad

$$[\![S \lhd P, \downarrow, \mathsf{o}, \oplus]\!]^{\mathrm{dc}} \stackrel{\mathsf{def}}{=} (D, \varepsilon, \delta)$$

where

  - $D : \mathbf{Set} \longrightarrow \mathbf{Set}$

    $D\,X \stackrel{\mathsf{def}}{=} \Sigma s : S.\,(P\,s \to X)$

  - $\varepsilon_X : D\,X \longrightarrow X$

    $\varepsilon_X\,(s, v) \stackrel{\mathsf{def}}{=} v\,(\mathsf{o}_{\{s\}})$

  - $\delta_X : D\,X \longrightarrow D\,D\,X$

    $\delta_X\,(s, v) \stackrel{\mathsf{def}}{=} \big(s, \lambda p.\,\big(s \downarrow p, \lambda p'.\,v\,(p \oplus_{\{s\}} p')\big)\big)$

Interpretation of dcon. morphisms

- Any directed container morphism

  $$t \lhd q : (S \lhd P, \downarrow, \text{o}, \oplus) \longrightarrow (S' \lhd P', \downarrow', \text{o}', \oplus')$$

  defines a natural transformation/~~comonad morphism~~

  $$\llbracket t \lhd q \rrbracket^{\mathrm{dc}} : \llbracket S \lhd P, \downarrow, \text{o}, \oplus \rrbracket^{\mathrm{dc}} \longrightarrow \llbracket S' \lhd P', \downarrow', \text{o}', \oplus' \rrbracket^{\mathrm{dc}}$$

  by

  - $\llbracket t \lhd q \rrbracket^{\mathrm{dc}}_X : \Sigma s : S. \, (P\, s \to X) \longrightarrow \Sigma s' : S'. \, (P'\, s' \to X)$

    $\llbracket t \lhd q \rrbracket^{\mathrm{dc}}_X \, (s, v) \overset{\text{def}}{=} (t\, s, v \circ q_{\{s\}})$

- $\llbracket - \rrbracket^{\mathrm{dc}}$ preserves the identities and composition

- $\llbracket - \rrbracket^{\mathrm{dc}}$ is a functor from **DCont** to $[\mathbf{Set}, \mathbf{Set}]$/$\cancel{\mathit{Comonads}(\mathit{Set})}$

Interpretation of dcon. morphisms

- Any directed container morphism

$$t \lhd q : (S \lhd P, \downarrow, \mathsf{o}, \oplus) \longrightarrow (S' \lhd P', \downarrow', \mathsf{o}', \oplus')$$

  defines a ~~natural transformation~~/comonad morphism

$$[\![t \lhd q]\!]^{\mathrm{dc}} : [\![S \lhd P, \downarrow, \mathsf{o}, \oplus]\!]^{\mathrm{dc}} \longrightarrow [\![S' \lhd P', \downarrow', \mathsf{o}', \oplus']\!]^{\mathrm{dc}}$$

  by

  - $[\![t \lhd q]\!]^{\mathrm{dc}}_X : \Sigma s : S.\,(P\, s \to X) \longrightarrow \Sigma s' : S'.\,(P'\, s' \to X)$

    $[\![t \lhd q]\!]^{\mathrm{dc}}_X\,(s, v) \stackrel{\mathrm{def}}{=} (t\, s, v \circ q_{\{s\}})$

- $[\![-]\!]^{\mathrm{dc}}$ preserves the identities and composition

- $[\![-]\!]^{\mathrm{dc}}$ is a functor from **DCont** to $\cancel{[\mathbf{Set}, \mathbf{Set}]}$/**Comonads**(**Set**)

Interpretation is fully faithful

- Every natural transformation/~~comonad morphism~~

$$\tau : [\![S \lhd P, \downarrow, \mathrm{o}, \oplus]\!]^{\mathrm{dc}} \longrightarrow [\![S' \lhd P', \downarrow', \mathrm{o}', \oplus']\!]^{\mathrm{dc}}$$

  defines a directed container morphism

$$\ulcorner \tau \urcorner^{\mathrm{dc}} : (S \lhd P, \downarrow, \mathrm{o}, \oplus) \longrightarrow (S' \lhd P', \downarrow', \mathrm{o}', \oplus')$$

  satisfying

  - $\ulcorner [\![t \lhd q]\!]^{\mathrm{dc}} \urcorner^{\mathrm{dc}} = t \lhd q$
  - $[\![\ulcorner \tau \urcorner^{\mathrm{dc}}]\!]^{\mathrm{dc}} = \tau$

- $[\![-]\!]^{\mathrm{dc}}$ is a fully faithful functor

Interpretation is fully faithful

- Every ~~natural transformation~~/comonad morphism

  $$\tau : [\![S \lhd P, \downarrow, \mathsf{o}, \oplus]\!]^{\mathrm{dc}} \longrightarrow [\![S' \lhd P', \downarrow', \mathsf{o}', \oplus']\!]^{\mathrm{dc}}$$

  defines a directed container morphism

  $$\ulcorner \tau \urcorner^{\mathrm{dc}} : (S \lhd P, \downarrow, \mathsf{o}, \oplus) \longrightarrow (S' \lhd P', \downarrow', \mathsf{o}', \oplus')$$

  satisfying

  - $\ulcorner [\![t \lhd q]\!]^{\mathrm{dc}} \urcorner^{\mathrm{dc}} = t \lhd q$
  - $[\![\ulcorner \tau \urcorner^{\mathrm{dc}}]\!]^{\mathrm{dc}} = \tau$

- $[\![-]\!]^{\mathrm{dc}}$ is a fully faithful functor

Directed containers = cons. $\cap$ cmnds.

- Any comonad (D, ε, δ), such that $D = [\![S \lhd P]\!]^{\mathrm{c}}$, determines

$$\lceil (D, \varepsilon, \delta), S \lhd P \rceil \stackrel{\mathsf{def}}{=} (S \lhd P, \downarrow, \mathsf{o}, \oplus)$$

- $\lceil - \rceil$ satisfies

$$[\![\lceil (D, \varepsilon, \delta), S \lhd P \rceil]\!]^{\mathrm{dc}} = (D, \varepsilon, \delta)$$

$$\lceil [\![S \lhd P, \downarrow, \mathsf{o}, \oplus]\!]^{\mathrm{dc}}, S \lhd P \rceil = (S \lhd P, \downarrow, \mathsf{o}, \oplus)$$

Directed containers = cons. ∩ cmnds.

- Any comonad (D, ε, δ), such that $D = [\![S \lhd P]\!]^{\mathrm{c}}$, determines

$$\lceil (D, \varepsilon, \delta), S \lhd P \rceil \stackrel{\mathsf{def}}{=} (S \lhd P, \downarrow, \mathsf{o}, \oplus)$$

- $\lceil - \rceil$ satisfies

$$[\![\lceil (D, \varepsilon, \delta), S \lhd P \rceil]\!]^{\mathrm{dc}} = (D, \varepsilon, \delta)$$

$$\lceil [\![S \lhd P, \downarrow, \mathsf{o}, \oplus]\!]^{\mathrm{dc}}, S \lhd P \rceil = (S \lhd P, \downarrow, \mathsf{o}, \oplus)$$

- The following is a pullback in **CAT**:

$$\begin{tikzcd}
\mathbf{DCont} \arrow[r, "U"] \arrow[d, "{[\![-]\!]^{\mathrm{dc}}}"', "\mathrm{f.f.}"] & \mathbf{Cont} \arrow[d, "\mathrm{f.f.}"', "{[\![-]\!]^{\mathrm{c}}}"] \\
\mathbf{Comonads}(\mathbf{Set}) \arrow[r, "U"] & {[\mathbf{Set}, \mathbf{Set}]}
\end{tikzcd}$$

Some constructions on directed containers

Coproducts of directed containers

- Given $(S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ and $(S_1 \lhd P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$, their coproduct is $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$ where

  - $S \lhd P = (S_0 \lhd P_0) + (S_1 \lhd P_1) = \left(S_0 + S_1 \lhd \left[\lambda s.\, P_0\, s, \lambda s.\, P_1\, s\right]\right)$

  - $\mathsf{inl}\, s \downarrow p = \mathsf{inl}\,(s \downarrow_0 p)$
    $\mathsf{inr}\, s \downarrow p = \mathsf{inr}\,(s \downarrow_1 p)$

  - $\mathsf{o}_{\{\mathsf{inl}\, s\}} = \mathsf{o}_{0\{s\}}$
    $\mathsf{o}_{\{\mathsf{inr}\, s\}} = \mathsf{o}_{1\{s\}}$

  - $p \oplus_{\{\mathsf{inl}\, s\}} p' = p \oplus_{0\,\{s\}} p'$
    $p \oplus_{\{\mathsf{inr}\, s\}} p' = p \oplus_{1\,\{s\}} p'$

Coproducts of directed containers

- Given $(S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ and $(S_1 \lhd P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$, their coproduct is $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$ where
  - $S \lhd P = (S_0 \lhd P_0) + (S_1 \lhd P_1) = \big(S_0 + S_1 \lhd \big[\lambda s.\, P_0\, s, \lambda s.\, P_1\, s\big]\big)$
  - $\mathsf{inl}\, s \downarrow p = \mathsf{inl}\, (s \downarrow_0 p)$
    $\mathsf{inr}\, s \downarrow p = \mathsf{inr}\, (s \downarrow_1 p)$
  - $\mathsf{o}_{\{\mathsf{inl}\, s\}} = \mathsf{o}_{0\,\{s\}}$
    $\mathsf{o}_{\{\mathsf{inr}\, s\}} = \mathsf{o}_{1\,\{s\}}$
  - $p \oplus_{\{\mathsf{inl}\, s\}} p' = p \oplus_{0\,\{s\}} p'$
    $p \oplus_{\{\mathsf{inr}\, s\}} p' = p \oplus_{1\,\{s\}} p'$
- It interprets as $[\![S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0]\!]^{\mathrm{dc}} + [\![S_1 \lhd P_1, \downarrow_1, \mathsf{o}_1, \oplus_1]\!]^{\mathrm{dc}}$

Products of strict directed containers

- Given $(S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ and $(S_1 \lhd P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$, there is no general way to endow $(S_0 \lhd P_0) \times (S_1 \lhd P_1)$ with dcon. struct.

Products of strict directed containers

- Given $(S_0 \lhd P_0, \downarrow_0, \mathsf{o}_0, \oplus_0)$ and $(S_1 \lhd P_1, \downarrow_1, \mathsf{o}_1, \oplus_1)$, there is no general way to endow $(S_0 \lhd P_0) \times (S_1 \lhd P_1)$ with dcon. struct.

- But analogously to (ideal) monads, the product exists for strict directed containers/coideal comonads:
  - S : **Set**
  - $P^+ : S \to$ **Set**
  - $\downarrow^+ : \Pi\, s : S.\, P^+\, s \to S$
  - $\oplus^+ : \Pi\, \{s : S\}.\, \Pi\, p : P^+\, s.\, P^+\, (s \downarrow^+ p) \to P^+\, s$
  - satisfying two laws (omitted)

- The dcon. determined by a strict dcon. has
  - $P\, s = 1 + P^+\, s$
  - ...

Products of strict directed containers ctd.

- Now, given $(S_0 \lhd P_0^+, \downarrow_0^+, \oplus_0^+)$ and $(S_1 \lhd P_1^+, \downarrow_1^+, \oplus_1^+)$, we can define $(S \lhd P^+, \downarrow^+, \oplus^+)$ where
  - $S = \overline{S_0} \times \overline{S_1}$
    with
    $(\overline{S_0}, \overline{S_1}) = \nu(Z_0, Z_1).\,\big(\Sigma s_0 : S_0.\, P_0^+\, s_0 \to Z_1\,,\, \Sigma s_1 : S_1.\, P_1^+\, s_1 \to Z_0\big)$
  - $P^+\,(s_0, s_1) = \overline{P_0^+\, s_0} + \overline{P_1^+\, s_1}$
    with
    $$\begin{aligned}(\overline{P_0^+\, s_0}, \overline{P_1^+\, s_1}) = \mu(Z_0, Z_1).\,\big(&\lambda(s_0, v_0).\,\Sigma p_0 : P_0^+\, s_0.1 + Z_1\,(v_0\; p_0)\,,\\ &\lambda(s_1, v_1).\,\Sigma p_1 : P_1^+\, s_1.1 + Z_0\,(v_1\; p_1)\big)\end{aligned}$$
  - ...

Products of strict directed containers ctd.

- Now, given $(S_0 \lhd P_0^+, \downarrow_0^+, \oplus_0^+)$ and $(S_1 \lhd P_1^+, \downarrow_1^+, \oplus_1^+)$, we can define $(S \lhd P^+, \downarrow^+, \oplus^+)$ where
  - $S = \overline{S_0} \times \overline{S_1}$
    with
    $(\overline{S_0}, \overline{S_1}) = \nu(Z_0, Z_1).\, \left(\Sigma s_0 : S_0.\, P_0^+\, s_0 \to Z_1\, ,\, \Sigma s_1 : S_1.\, P_1^+\, s_1 \to Z_0\right)$
  - $P^+\,(s_0, s_1) = \overline{P_0^+\, s_0} + \overline{P_1^+\, s_1}$
    with
    $$(\overline{P_0^+\, s_0}, \overline{P_1^+\, s_1}) = \mu(Z_0, Z_1).\, \big(\lambda(s_0, v_0).\, \Sigma p_0 : P_0^+\, s_0.1 + Z_1\,(v_0\, p_0)\, ,\\ \lambda(s_1, v_1).\, \Sigma p_1 : P_1^+\, s_1.1 + Z_0\,(v_1\, p_1)\big)$$
  - ...
- This gives the product of the given strict dcons. in **DCont**
- It interprets as a product of the corresponding coideal comonads

Focussing a container

- Given any container $S_0 \lhd P_0$, we get $(S \lhd P, \downarrow, \mathrm{o}, \oplus)$ where
  - $S = \Sigma s : S_0 . P_0\, s$
  - $P\,(s, p) = P_0\, s$
  - $(s, p) \downarrow p' = (s, p')$
  - $\mathrm{o}_{\{s,p\}} = p$
  - $p' \oplus_{\{s,p\}} p'' = p''$

Focussing a container

- Given any container $S_0 \lhd P_0$, we get $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$ where
  - $S = \Sigma s : S_0.P_0\, s$
  - $P\,(s, p) = P_0\, s$
  - $(s, p) \downarrow p' = (s, p')$
  - $\mathsf{o}_{\{s,p\}} = p$
  - $p' \oplus_{\{s,p\}} p'' = p''$
- This dcon. interprets into a comonad on $\partial[\![S_0 \lhd P_0]\!]^{\mathrm{c}} \times \mathsf{Id}$
- Focussing forms a functor from $\mathbf{Con}_{\mathsf{cart}}$ to $\mathbf{DCon}$

Other constructions on directed containers

- Cofree and cofree recursive directed containers

Other constructions on directed containers

- Cofree and cofree recursive directed containers
- Distributive laws between directed containers
  - $t^\theta \lhd q^\theta : (S_0 \lhd P_0) \circ^{\mathrm{c}} (S_1 \lhd P_1) \longrightarrow (S_1 \lhd P_1) \circ^{\mathrm{c}} (S_0 \lhd P_0)$ satisfying 11 laws (and with $t_0^\theta\,(s, v) = v\,(\mathrm{o}_{0\{s\}})$ forced)

Other constructions on directed containers

- Cofree and cofree recursive directed containers

- Distributive laws between directed containers
  - $t^{\theta} \lhd q^{\theta} : (S_0 \lhd P_0) \circ^{\mathrm{c}} (S_1 \lhd P_1) \longrightarrow (S_1 \lhd P_1) \circ^{\mathrm{c}} (S_0 \lhd P_0)$ satisfying 11 laws (and with $t_0^{\theta}\,(s, v) = v\,(\mathrm{o}_{0\{s\}})$ forced)
  - A dep. typed version of the Zappa–Szép product, i.e., of:
    - Given monoid actions $\alpha : N \times M \to M$ and $\beta : N \times M \to N$ satisfying two compat. laws, we get a monoid on $M \times N$ with $(m_0, n_0) \oplus (m_1, n_1) = \big(m_0 \oplus_M \alpha(n_0, m_1)\,,\, \beta(n_0, m_1) \oplus_N n_1\big)$

Other constructions on directed containers

- Cofree and cofree recursive directed containers
- Distributive laws between directed containers
  - $t^\theta \lhd q^\theta : (S_0 \lhd P_0) \circ^{\mathrm{c}} (S_1 \lhd P_1) \longrightarrow (S_1 \lhd P_1) \circ^{\mathrm{c}} (S_0 \lhd P_0)$ satisfying 11 laws (and with $t^\theta_0\,(s, v) = v\,(\mathsf{o}_{0\{s\}})$ forced)
  - A dep. typed version of the Zappa–Szép product, i.e., of:
    - Given monoid actions $\alpha : N \times M \to M$ and $\beta : N \times M \to N$ satisfying two compat. laws, we get a monoid on $M \times N$ with $(m_0, n_0) \oplus (m_1, n_1) = \big(m_0 \oplus_M \alpha(n_0, m_1)\,,\, \beta(n_0, m_1) \oplus_N n_1\big)$
- Composition of directed containers
  - Dist. laws give dcon. struct. on the comp. of underlying cons.
  - **Examples:** ne. lists over ne. lists, *streams over streams*, . . .

Other constructions on directed containers

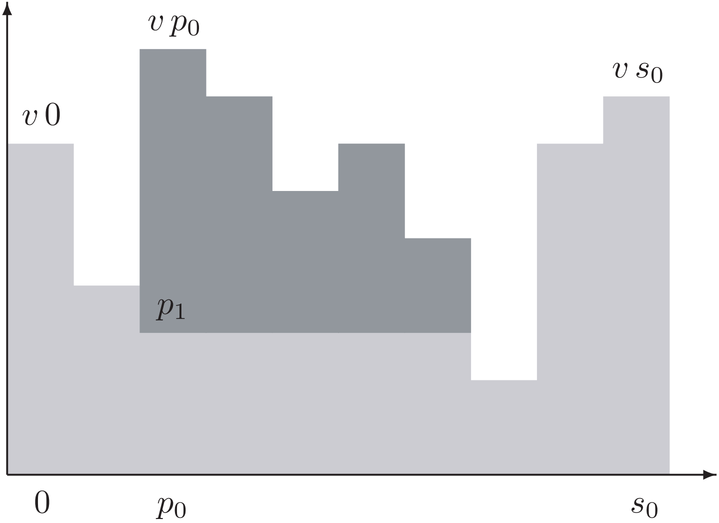
- Cofree and cofree recursive directed containers

- Distributive laws between directed containers
  - $t^\theta \lhd q^\theta : (S_0 \lhd P_0) \circ^{\mathrm{c}} (S_1 \lhd P_1) \longrightarrow (S_1 \lhd P_1) \circ^{\mathrm{c}} (S_0 \lhd P_0)$ satisfying 11 laws (and with $t_0^\theta\,(s, v) = v\,(\mathrm{o}_{0\{s\}})$ forced)
  - A dep. typed version of the Zappa–Szép product, i.e., of:
    - Given monoid actions $\alpha : N \times M \to M$ and $\beta : N \times M \to N$ satisfying two compat. laws, we get a monoid on $M \times N$ with $(m_0, n_0) \oplus (m_1, n_1) = \big(m_0 \oplus_M \alpha(n_0, m_1)\,,\, \beta(n_0, m_1) \oplus_N n_1\big)$

- Composition of directed containers
  - Dist. laws give dcon. struct. on the comp. of underlying cons.
  - **Examples:** ne. lists over ne. lists, *streams over streams*, ...

“This should be called an aqueduct” —A.M.Pitts

Non-empty lists over non-empty lists



Directed containers and effects

Directed containers and effects

- **Recall:** Given a directed container $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$, we get a comonad on $D\,X = \Sigma s : S.\,(P\,s \to X)$

Directed containers and effects

- **Recall:** Given a directed container $(S \lhd P, \downarrow, \mathrm{o}, \oplus)$, we get a comonad on $D\,X = \Sigma s : S.\,(P\,s \to X)$

- If you stare long enough at it (and it's quite late), then

Directed containers and effects

- **Recall:** Given a directed container $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$, we get a comonad on $D\,X = \Sigma s : S.\,(P\,s \to X)$

- If you stare long enough at it (and it's quite late), then
  - it looks a lot like the costate comonad on $S \times (S \to -)$

Directed containers and effects

- **Recall:** Given a directed container $(S \lhd P, \downarrow, \mathrm{o}, \oplus)$, we get a comonad on $D\,X = \Sigma s : S.\,(P\,s \to X)$
- If you stare long enough at it (and it's quite late), then
  - it looks a lot like the costate comonad on $S \times (S \to -)$
  - what happens if instead we take $T\,X = \Pi s : S.\,(P\,s \times X)$?

Directed containers and effects

- **Recall:** Given a directed container $(S \lhd P, \downarrow, o, \oplus)$, we get a comonad on $D\,X = \Sigma s : S.\,(P\,s \to X)$

- If you stare long enough at it (and it's quite late), then
  - it looks a lot like the costate comonad on $S \times (S \to -)$
  - what happens if instead we take $T\,X = \Pi s : S.\,(P\,s \times X)$?
  - it looks suspiciously like the state monad $S \to (S \times -)$

Cointerpretation of (directed) containers

- In addition to the interpretation functor

$$[\![-]\!]^{\mathrm{c}} : \mathbf{Cont} \longrightarrow [\mathbf{Set}, \mathbf{Set}]$$

one can also define a cointerpretation functor

$$\langle\!\langle - \rangle\!\rangle^{\mathrm{c}} : \mathbf{Cont}^{\mathrm{op}} \longrightarrow [\mathbf{Set}, \mathbf{Set}]$$

given by

$$\langle\!\langle S \lhd P \rangle\!\rangle^{\mathrm{c}}\, X \stackrel{\mathrm{def}}{=} \Pi s : S.\, (P\, s \times X)$$

which lifts to $\langle\!\langle - \rangle\!\rangle^{\mathrm{dc}}$, making the following a pullback in **CAT**

$$\begin{tikzcd} \mathbf{DCont}^{\mathrm{op}} \arrow[r, "U"] \arrow[d, "\langle\!\langle - \rangle\!\rangle^{\mathrm{dc}}"'] & \mathbf{Cont}^{\mathrm{op}} \arrow[d, "\langle\!\langle - \rangle\!\rangle^{\mathrm{c}}"] \\ \mathbf{Monads}(\mathbf{Set}) \arrow[r, "U"] & [\mathbf{Set}, \mathbf{Set}] \end{tikzcd}$$

Dependently typed update monads

- In more detail, given a directed container $(S \lhd P, \downarrow, \mathrm{o}, \oplus)$, the corresponding dependently typed update monad is given by
  - $T : \mathbf{Set} \longrightarrow \mathbf{Set}$
    $T\,X \stackrel{\text{def}}{=} \langle\!\langle S \lhd P \rangle\!\rangle^{\mathrm{c}}\,X = \Pi s : S.\,(P\,s \times X)$
  - $\eta_X : X \longrightarrow T\,X$
    $\eta_X\,x \stackrel{\text{def}}{=} \lambda s.\,(\mathrm{o}_{\{s\}}, x)$
  - $\mu_X : T\,T\,X \longrightarrow T\,X$
    $$\begin{aligned} \mu_X\,f \stackrel{\text{def}}{=} \lambda s.\, &\mathbf{let}\ (p, g) = f\,s\ \mathbf{in} \\ &\mathbf{let}\ (p', x) = g\,(s \downarrow p)\ \mathbf{in}\ (p \oplus_{\{s\}} p', x) \end{aligned}$$
- Intuitively
  - S – set/type of states
  - $(P, \mathrm{o}, \oplus)$ – dependently typed monoid of state updates

Dependently typed update monads ctd.

- The dependently typed update monad

$$T\,X \stackrel{\text{def}}{=} \Pi s : S.\,(P\,s \times X)$$

arises as the free-model monad for a (large) Lawvere theory, whose models are given by a carrier M : **Set** and two operations

$$\mathsf{lkp} : (S \to M) \longrightarrow M \qquad \mathsf{upd} : (\Pi s : S.\,P\,s) \times M \longrightarrow M$$

subject to three natural equations

  - $\mathsf{lkp}\,(\lambda s.\,\mathsf{upd}_{\lambda s.\,o_{\{s\}}}(m)) = m$
  - $\mathsf{lkp}\,(\lambda s.\,\mathsf{upd}_f\,(\mathsf{lkp}\,(\lambda s'.\,m\,s'))) = \mathsf{lkp}(\lambda s.\,\mathsf{upd}_f\,(m\,(s \downarrow (f\,s))))$
  - $\mathsf{upd}_f\,(\mathsf{upd}_g\,(m)) = \mathsf{upd}_{\lambda s.\,(f\,s) \oplus (g\,(s \downarrow f\,s))}\,(m)$

Examples of dep. typed update monads

- Global state
  - S : **Set**
  - $P\,s = S$

  - $s \downarrow s' = s'$
  - $\mathsf{o}_{\{s\}} = s$
  - $s' \oplus_{\{s\}} s'' = s''$

  - $T\,X = S \to (S \times X)$

Examples of dep. typed update monads ctd.

- Monotonic state as in F*
  - S : **Set**
  - $P\,s = \{s' : S \mid s \mathrel{\mathcal{R}} s'\}$

    where $\mathcal{R}$ is some fixed preorder on S, e.g.,
    - $\leq$ when $S = \mathsf{Nat}$ and modelling monotonic counters
    - transition relation of some state machine
    - subset relation for references when $S = \mathsf{heap}$
  - $s \downarrow s' = s'$
  - $\mathsf{o}_{\{s\}} = s$
  - $s' \oplus_{\{s\}} s'' = s''$
  - $T\,X = \Pi s : S.\,(\{s' : S \mid s \mathrel{\mathcal{R}} s'\} \times X)$
  - In F* it is combined with a modal logic based Hoare logic

Examples of dep. typed update monads ctd.

- A non-overflowing (non-removal) buffer
  - fixed sized buffer of length n
  - storing values of some type A

  - $S = A^{\leq n}$
  - $P\, as = A^{\leq n - \mathsf{len}\, as}$

  - $as \downarrow as' = as +\!\!+ as'$
  - $\mathsf{o}_{\{as\}} = []$
  - $as' \oplus_{\{as\}} as'' = as' +\!\!+ as''$

  - $T\, X = \Pi as : A^{\leq n}.\, (A^{\leq n - \mathsf{len}\, as} \times X)$

Examples of dep. typed update monads ctd.

- A non-underflowing (unbounded) stack
  - $S = A^*$
  - $P\,as = \{ps : (1 + A)^* \mid \mathrm{removes}\,ps \leq \mathrm{len}\,as\}$

    where

    $$\begin{aligned}
    \mathrm{removes}\,[] &= 0 \\
    \mathrm{removes}\,(\mathrm{inl}\,* :: ps) &= \mathrm{removes}\,ps + 1 \\
    \mathrm{removes}\,(\mathrm{inr}\,a :: ps) &= \mathrm{removes}\,ps \dot{-} 1
    \end{aligned}$$

  - $as \downarrow [] = as$

    $as \downarrow (\mathrm{inl}\,* :: ps) = as/1 \downarrow ps$

    $as \downarrow (\mathrm{inr}\,a :: ps) = (as \mathbin{+\!\!+} [a]) \downarrow ps$
  - $\mathrm{o}_{\{as\}} = []$
  - $as' \oplus_{\{as\}} as'' = as' \mathbin{+\!\!+} as''$

Simply typed update monads

- If P : **Set**, then we get a simply typed update monad

$$T\,X \stackrel{\text{def}}{=} S \to (P \times X)$$

- In this case,
  - $(P, \mathsf{o}, \oplus)$ is a monoid in the standard sense
  - $\downarrow : S \times P \longrightarrow S$ is an action of $(P, \mathsf{o}, \oplus)$ on S

- This monad is the compatible composition of the monads

$$T_{\text{reader}}\,X \stackrel{\text{def}}{=} S \to X \qquad T_{\text{writer}}\,X \stackrel{\text{def}}{=} P \times X$$

- There is a one-to-one correspondence between
  - monoid actions $\downarrow : S \times P \longrightarrow S$
  - distributive laws $\theta : T_{\text{writer}} \circ T_{\text{reader}} \longrightarrow T_{\text{reader}} \circ T_{\text{writer}}$

Directed containers and BX

Directed containers and BX

- An asymmetric lens is a comodel for the th. of global state, i.e.,
  - X : **Set** (the database)
  - get : $X \longrightarrow S$ (computing the view)
  - put : $X \times S \longrightarrow X$ (updating the database)
  - satisfying natural laws relating get and put
- Equivalently a coalgebra for the costate comonad $S \times (S \to -)$

Directed containers and BX

- An asymmetric lens is a comodel for the th. of global state, i.e.,
  - X : **Set** (the database)
  - get : $X \longrightarrow S$ (computing the view)
  - put : $X \times S \longrightarrow X$ (updating the database)
  - satisfying natural laws relating get and put
- Equivalently a coalgebra for the costate comonad $S \times (S \to -)$
- Given a simply typed dcon. $(S \lhd P, \downarrow, \mathrm{o}, \oplus)$, i.e., where P : **Set**, we define a simply typed update lens to be given by
  - X : **Set**
  - lkp : $X \longrightarrow S$
  - upd : $X \times P \longrightarrow X$
  - satisfying natural laws relating lkp and upd
- Equivalently a coalgebra for $[\![S \lhd P, \downarrow, \mathrm{o}, \oplus]\!]^{\mathrm{dc}}$

Directed containers and BX ctd.

- Analogously, given a general dcon. $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$,
  we can define a dependently typed update lens to be given by
  - X : **Set**
  - $\mathsf{lkp} : X \longrightarrow S$
  - $\mathsf{upd} : \big(\Sigma x : X.P\,(\mathsf{lkp}\,x)\big) \longrightarrow X$
  - satisfying natural laws relating lkp and upd

Directed containers and BX ctd.

- Analogously, given a general dcon. $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$, we can define a dependently typed update lens to be given by
  - X : **Set**
  - $\mathsf{lkp} : X \longrightarrow S$
  - $\mathsf{upd} : \big(\Sigma x : X . P\,(\mathsf{lkp}\,x)\big) \longrightarrow X$
  - satisfying natural laws relating lkp and upd

- Previous examples were about asymmetric update lenses, but it is also possible to do a more symmetric variant with dcons.:
  - $$\mathsf{fwd} \lhd \mathsf{bwd} : (S_{\mathsf{db}} \lhd P_{\mathsf{db}}, \downarrow_{\mathsf{db}}, \mathsf{o}_{\mathsf{db}}, \oplus_{\mathsf{db}}) \longrightarrow (S_{\mathsf{view}} \lhd P_{\mathsf{view}}, \downarrow_{\mathsf{view}}, \mathsf{o}_{\mathsf{view}}, \oplus_{\mathsf{view}})$$
  - now both the database and the view have their own updates

Directed containers and (small) categories

Directed containers and (small) categories

- Given a directed container $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$ we get a corresponding small category $\mathcal{C}_{(S \lhd P, \downarrow, \mathsf{o}, \oplus)}$ as follows
  - $\mathrm{ob}(\mathcal{C}) \stackrel{\mathrm{def}}{=} S$
  - $\mathcal{C}(s, s') \stackrel{\mathrm{def}}{=} \Sigma p : P\, s.\, (s \downarrow p = s')$
  - identities are given using o
  - composition is given using $\oplus$
- And vice versa, every small category $\mathcal{C}$ gives us a corresponding directed container $(S_{\mathcal{C}} \lhd P_{\mathcal{C}}, \downarrow_{\mathcal{C}}, \mathsf{o}_{\mathcal{C}}, \oplus_{\mathcal{C}})$
- But then, is it simply the case that **Cat** $\cong$ **DCont**?

Directed containers and (small) categories

- Given a directed container $(S \lhd P, \downarrow, \mathsf{o}, \oplus)$ we get a corresponding small category $\mathcal{C}_{(S \lhd P, \downarrow, \mathsf{o}, \oplus)}$ as follows
    - $\mathrm{ob}(\mathcal{C}) \stackrel{\text{def}}{=} S$
    - $\mathcal{C}(s, s') \stackrel{\text{def}}{=} \Sigma p : P\, s.\, (s \downarrow p = s')$
    - identities are given using o
    - composition is given using $\oplus$
- And vice versa, every small category $\mathcal{C}$ gives us a corresponding directed container $(S_{\mathcal{C}} \lhd P_{\mathcal{C}}, \downarrow_{\mathcal{C}}, \mathsf{o}_{\mathcal{C}}, \oplus_{\mathcal{C}})$
- But then, is it simply the case that **Cat** $\cong$ **DCont**? NO!

Directed container morphisms as cofunctors

- Given a directed container morphism

$$t \lhd q : (S \lhd P, \downarrow, \mathsf{o}, \oplus) \longrightarrow (S' \lhd P', \downarrow', \mathsf{o}', \oplus')$$

we do not get a functor, but instead a cofunctor [Aguiar'97]

$$F_{t \lhd q} : \mathcal{C}_{(S \lhd P, \downarrow, \mathsf{o}, \oplus)} \longrightarrow \mathcal{D}_{(S' \lhd P', \downarrow', \mathsf{o}', \oplus')}$$

given by a mapping of objects

$$(F_{t \lhd q})_0 \stackrel{\text{def}}{=} t : \mathrm{ob}(\mathcal{C}) \longrightarrow \mathrm{ob}(\mathcal{D})$$

and a lifting operation on morphisms (pre-opcleavage)

$$\begin{tikzcd}
s \arrow[r, "(F_{t \lhd q})_1(s,p) \stackrel{\text{def}}{=} q_{\{s\}}\, p"] & \circledast & \text{in } \mathcal{C} \\
(F_{t \lhd q})_0(s) \arrow[r, "p"'] \arrow[u, phantom, "\uparrow"] & s' & \text{in } \mathcal{D}
\end{tikzcd}$$

Constructions on dcons. revisited

- On the one hand, we can relate existing constructions on directed containers to constructions (small) categories, e.g.,
  - the symmetry of the definition of directed polynomials in

    $$\mathsf{s} : \overline{P} \longrightarrow S \qquad \text{and} \qquad \downarrow : \overline{P} \longrightarrow S$$

    manifests as every category having an opposite category

Constructions on dcons. revisited

- On the one hand, we can relate existing constructions on directed containers to constructions (small) categories, e.g.,
  - the symmetry of the definition of directed polynomials in

    $$\mathsf{s} : \overline{P} \longrightarrow S \qquad \text{and} \qquad \downarrow : \overline{P} \longrightarrow S$$

    manifests as every category having an opposite category

  - bidirected containers (ongoing, a bit of a conundrum) with

    $$(-)^{-1} : \Pi\{s : S\}.\, \Pi p : P\, s.\, P\,(s \downarrow p)$$

    then correspond to groupoids

Constructions on dcons. revisited

- On the one hand, we can relate existing constructions on directed containers to constructions (small) categories, e.g.,
  - the symmetry of the definition of directed polynomials in

    $$\mathsf{s} : \overline{P} \longrightarrow S \qquad \text{and} \qquad \downarrow : \overline{P} \longrightarrow S$$

    manifests as every category having an opposite category
  - bidirected containers (ongoing, a bit of a conundrum) with

    $$(-)^{-1} : \Pi\{s : S\}. \Pi p : P\, s. P\,(s \downarrow p)$$

    then correspond to groupoids
- On the other hand, the (small) categories view also provides new insights into directed containers and comonads, e.g.,
  - factorisation of directed container/comonad morphisms

Factorisation of morphisms

- Given a directed container morphism

$$t \lhd q : (S \lhd P, \downarrow, \mathsf{o}, \oplus) \longrightarrow (S' \lhd P', \downarrow', \mathsf{o}', \oplus')$$

we can factorise $(t \lhd q)$ as $(t \lhd \lambda s.\, \mathrm{id}_{P'(t\,s)}) \circ (\mathrm{id}_S \lhd q)$ where

in **Cont**

$$\begin{tikzcd}
(S \lhd P) \arrow[r, dotted, "\mathrm{id}_S \lhd q"] \arrow[rr, bend left, "t \lhd q"] \arrow[dr, bend right, "\mathrm{id}_S \lhd \lambda _.?"'] & (S \lhd P' \circ t) \arrow[r, "t \lhd \lambda s.\mathrm{id}_{P'(t\,s)}"] \arrow[d, "\mathrm{id}_S \lhd \lambda _.?"'] \arrow[dr, phantom, "\mathrm{pb.}"] & (S' \lhd P') \arrow[d, "\mathrm{id}_{S'} \lhd \lambda _.?"] \\
 & (S \lhd \lambda _.\, 0) \arrow[r, "t \lhd \lambda _.\mathrm{id}_0"'] & (S' \lhd \lambda _.\, 0)
\end{tikzcd}$$

inspired by the full image factorisation of ordinary functors

- Notably, this works for all pullback-preserving comonads

Conclusions

- Directed containers
  - type-theoretic and polynomial presentations
  - their use in functional programming
  - why are they canonical such structure?
- Some constructions on directed containers
  - coproducts of directed containers
  - strict directed containers and their products
  - focussing a container
  - ...
- Directed containers and computational effects
- Directed containers and BX
- Directed containers and categories