Comodule Representations of Second-Order Functionals

Danel Ahman (University of Tartu, Estonia)

Andrej Bauer (University of Ljubljana, Slovenia)

TYPES 2024

10–14 June 2024

Tree representations of continuous functionals

An **abstract view** of tree representations

$\boxed{\textbf{Comodule representations}}$

. . . . . .

Continuous functionals | ***Fin. supported*** functionals | ***Functional*** functionals | ***Exceptional*** functionals | ***Constant*** functionals | ***Interactive*** functionals | ***Partial*** functionals | ***Instance reductions*** | . . .

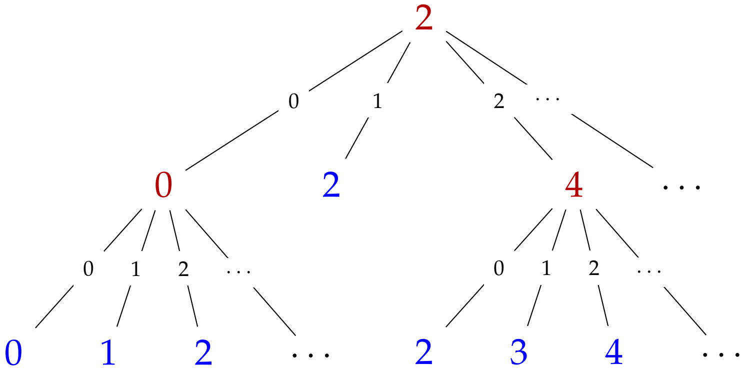
Tree representations of continuous functionals

Tree representations of continuous functionals

- Consider $F : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$, say $F(h) = h(2 \cdot h(2)) + h(2)$

Tree representations of continuous functionals

- Consider $F : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$, say $F(h) = h(2 \cdot h(2)) + h(2)$
- A ***tree representation*** of F:



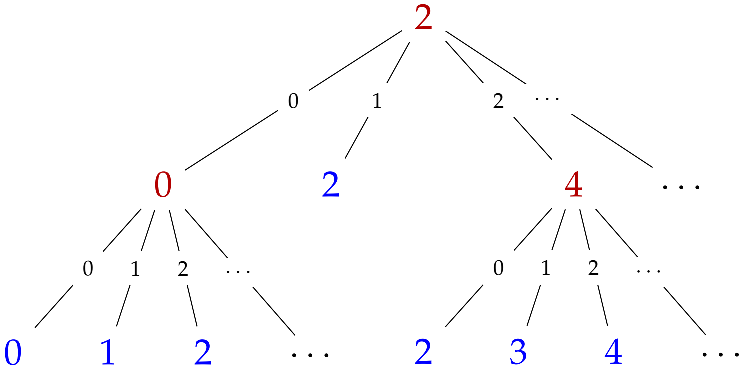
(a $\mathbb{N}$-labelled, $\mathbb{N}$-branching, $\mathbb{N}$-leaved well-founded tree)

Tree representations of continuous functionals

- Consider $F : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$, say $F(h) = h(2 \cdot h(2)) + h(2)$
- A ***tree representation*** of F:

$$h = (1, 5, 0, 3, 9, \ldots)$$

$$\begin{aligned} &F(h) \\ = \; &h(2 \cdot h(2)) + h(2) \end{aligned}$$



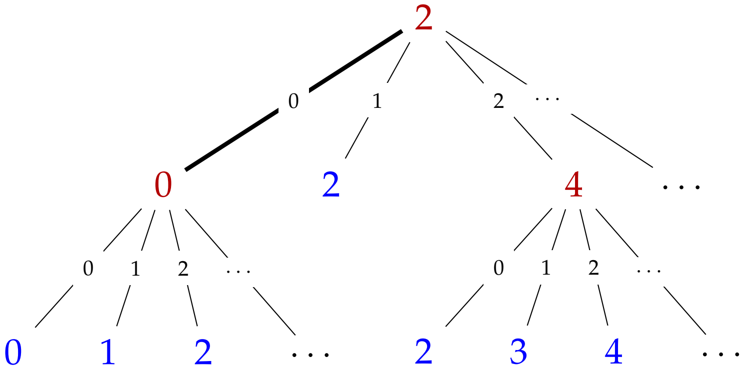
(a $\mathbb{N}$-labelled, $\mathbb{N}$-branching, $\mathbb{N}$-leaved well-founded tree)

Tree representations of continuous functionals

- Consider $F : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$, say $F(h) = h(2 \cdot h(2)) + h(2)$
- A ***tree representation*** of F:

$$h = (1, 5, 0, 3, 9, \ldots)$$

$$\begin{aligned} & F(h) \\ = \ & h(2 \cdot h(2)) + h(2) \\ = \ & h(2 \cdot 0) + 0 \end{aligned}$$



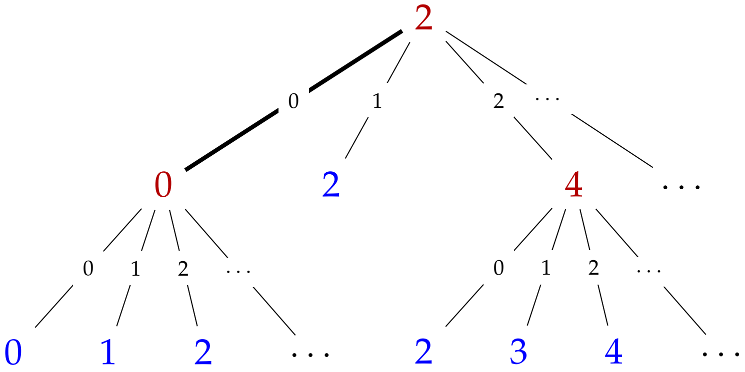
(a $\mathbb{N}$-labelled, $\mathbb{N}$-branching, $\mathbb{N}$-leaved well-founded tree)

Tree representations of continuous functionals

- Consider $F : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$, say $F(h) = h(2 \cdot h(2)) + h(2)$
- A ***tree representation*** of F:

$$h = (1, 5, 0, 3, 9, \ldots)$$

$$\begin{aligned} & F(h) \\ = \; & h(2 \cdot h(2)) + h(2) \\ = \; & h(2 \cdot 0) + 0 \\ = \; & h(0) + 0 \end{aligned}$$



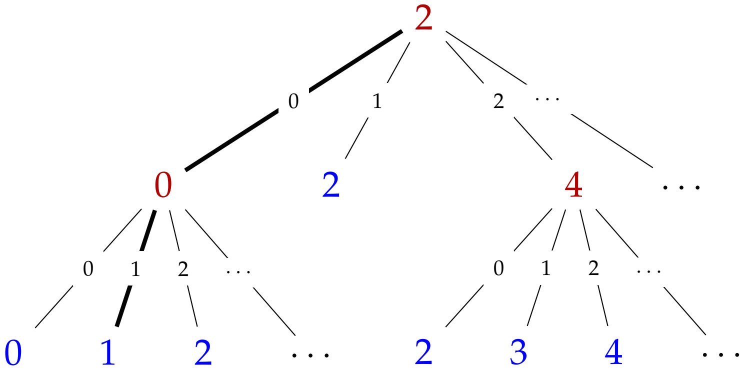
(a $\mathbb{N}$-labelled, $\mathbb{N}$-branching, $\mathbb{N}$-leaved well-founded tree)

Tree representations of continuous functionals

- Consider $F : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$, say $F(h) = h(2 \cdot h(2)) + h(2)$
- A ***tree representation*** of F:

$$h = (1, 5, 0, 3, 9, \ldots)$$

$$\begin{aligned} & F(h) \\ =\ & h(2 \cdot h(2)) + h(2) \\ =\ & h(2 \cdot 0) + 0 \\ =\ & h(0) + 0 \\ =\ & 1 + 0 \end{aligned}$$



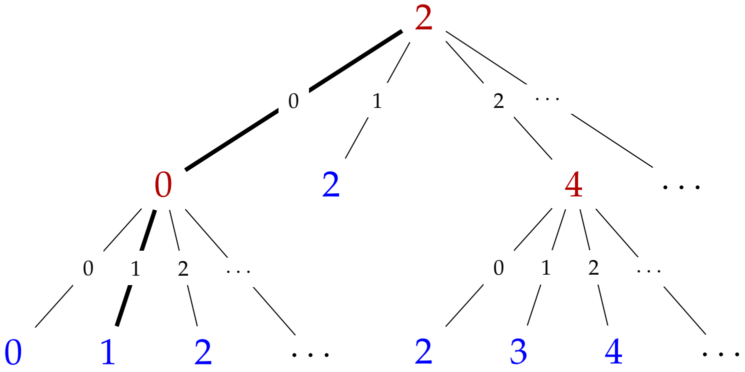
(a $\mathbb{N}$-labelled, $\mathbb{N}$-branching, $\mathbb{N}$-leaved well-founded tree)

Tree representations of continuous functionals

- Consider $F : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$, say $F(h) = h(2 \cdot h(2)) + h(2)$
- A ***tree representation*** of F:

$$h = (1, 5, 0, 3, 9, \ldots)$$

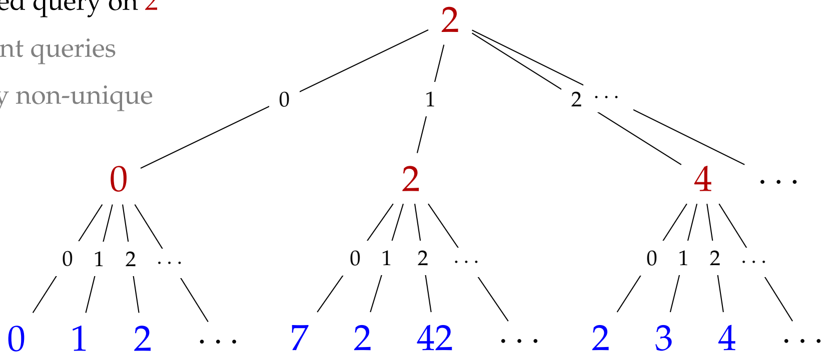
$$\begin{aligned} & F(h) \\ = \; & h(2 \cdot h(2)) + h(2) \\ = \; & h(2 \cdot 0) + 0 \\ = \; & h(0) + 0 \\ = \; & 1 + 0 \\ = \; & 1 \end{aligned}$$



(a $\mathbb{N}$-labelled, $\mathbb{N}$-branching, $\mathbb{N}$-leaved well-founded tree)

Tree representations of continuous functionals

- Consider $F : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$, say $F(h) = h(2 \cdot h(2)) + h(2)$
- ***Another tree representation*** of F:
  - duplicated query on 2
  - redundant queries
  - generally non-unique



(a $\mathbb{N}$-labelled, $\mathbb{N}$-branching, $\mathbb{N}$-leaved well-founded tree)

An **abstract view** of tree representations

An abstract view of tree representations: **trees**

- Given $A : \mathbf{Type}$ and $P : A \to \mathbf{Type}$
- ***Type of trees***: inductively define $\mathsf{Tree}(A, P)$ by

$$\frac{}{\mathsf{leaf} : \mathsf{Tree}(A, P)} \qquad \frac{a : A \qquad ts : P\,a \to \mathsf{Tree}(A, P)}{\mathsf{node}(a, ts) : \mathsf{Tree}(A, P)}$$

(A-labelled, P-branching wf. trees with **unlabelled leaves**)

An abstract view of tree representations: **trees and paths**

- Given $A : \mathbf{Type}$ and $P : A \to \mathbf{Type}$
- ***Type of trees***: inductively define $\mathsf{Tree}(A, P)$ by

$$\frac{}{\mathsf{leaf} : \mathsf{Tree}(A,P)} \qquad \frac{a : A \qquad ts : P\,a \to \mathsf{Tree}(A,P)}{\mathsf{node}(a, ts) : \mathsf{Tree}(A,P)}$$

(A-labelled, P-branching wf. trees with **<u>unlabelled</u> leaves**)

- ***Type of paths***: given $t : \mathsf{Tree}(A, P)$, inductively define $\mathsf{Path}_{A,P}(t)$ by

$$\frac{}{\mathsf{stop} : \mathsf{Path}_{A,P}(\mathsf{leaf})} \qquad \frac{p : P\,a \qquad \vec{p} : \mathsf{Path}_{A,P}(ts\,p)}{\mathsf{step}(p, \vec{p}) : \mathsf{Path}_{A,P}(\mathsf{node}(a, ts))}$$

(paths from root to leaves)

An abstract view of tree representations: **computing a path**

- Given

$$h : \prod_{a:A} P\,a \qquad \text{and} \qquad t : \mathsf{Tree}(A,P)$$

we can recursively ***compute a path*** $\mathsf{c}_{A,P}\,h\,t : \mathsf{Path}_{A,P}(t)$ by

$$\begin{aligned}
\mathsf{c}_{A,P}\,h\,\mathsf{leaf} &\stackrel{\text{def}}{=} \mathsf{stop} \\
\mathsf{c}_{A,P}\,h\,(\mathsf{node}(a,t)) &\stackrel{\text{def}}{=} \mathsf{step}\big(h\,a, \mathsf{c}_{A,P}\,h\,(t\,(h\,a))\big)
\end{aligned}$$

- This defines a map

$$\mathsf{c}_{A,P} : \left(\prod_{a:A} P\,a\right) \to \prod_{t:\mathsf{Tree}(A,P)} \mathsf{Path}_{A,P}(t)$$

An abstract view of tree representations: **tree representations**

- A ***tree representation*** of a (continuous) functional

$$F : \left(\prod_{a:A} P\,a\right) \to \left(\prod_{b:B} Q\,b\right)$$

consists of

An abstract view of tree representations: **tree representations**

- A ***tree representation*** of a (continuous) functional

$$F : \left(\prod_{a:A} P\,a\right) \to \left(\prod_{b:B} Q\,b\right)$$

consists of maps

$$\mathsf{t}_F : B \to \mathsf{Tree}(A,P) \quad \text{and} \quad \mathsf{e}_F : \prod_{\{b:B\}} \mathsf{Path}_{A,P}(\mathsf{t}_F\,b) \to Q\,b$$

such that the following diagram commutes:

$$\begin{tikzcd}
\prod_{a:A} P\,a \arrow[rr, "F"] \arrow[dr, bend right, "\mathsf{c}_{A,P}"'] & & \prod_{b:B} Q\,b \\
& \prod_{t:\mathsf{Tree}(A,P)} \mathsf{Path}_{A,P}(t) \arrow[ur, bend right, "\lambda\alpha.\,\lambda b.\,\mathsf{e}_F\,(\alpha\,(\mathsf{t}_F\,b))"'] &
\end{tikzcd}$$

$$\left(\,F\,h\,b = \mathsf{e}_F\,(\mathsf{c}_{A,P}\,h\,(\mathsf{t}_F\,b))\,\right)$$

Can this situation be captured **even more abstractly**?

$$
\begin{tikzcd}
\prod_{a:A} P\,a \arrow[rr, "F"] \arrow[dr, bend right, "\mathsf{c}_{A,P}"'] & & \prod_{b:B} Q\,b \\
& \prod_{t:\mathsf{Tree}(A,P)} \mathsf{Path}_{A,P}(t) \arrow[ur, bend right, "\lambda\alpha.\,\lambda b.\,\mathsf{e}_F\,(\alpha\,(\mathsf{t}_F\,b))"'] &
\end{tikzcd}
$$

Capturing tree representations more abstractly: **containers**

- A ***container*** $A \lhd P$ is given by:
  - a type A of ***shapes***, and
  - a family $P : A \to \mathbf{Type}$ of ***positions***
- Examples:
  - ***Lists:*** $\mathbb{N} \lhd \lambda n. \{0, 1, \ldots, n-1\}$
  - ***Trees:*** $\mathsf{Tree}(A, P) \lhd \lambda t. \mathsf{Path}_{A,P}(t)$

Capturing tree representations more abstractly: **containers**

- A ***container*** $A \lhd P$ is given by:
  - a type A of ***shapes***, and
  - a family $P : A \to \mathbf{Type}$ of ***positions***
- Examples:
  - ***Lists:*** $\mathbb{N} \lhd \lambda n.\{0, 1, \ldots, n-1\}$
  - ***Trees:*** $\mathsf{Tree}(A, P) \lhd \lambda t.\,\mathsf{Path}_{A,P}(t)$
- A ***container morphism*** $f \lhd g : (A \lhd P) \to (B \lhd Q)$ is given by

$$f : A \to B \qquad \text{and} \qquad g : \prod_{\{a:A\}} Q(f\,a) \to P\,a$$

Capturing tree representations more abstractly: **containers**

- A ***container*** $A \lhd P$ is given by:
  - a type A of ***shapes***, and
  - a family $P : A \to \mathbf{Type}$ of ***positions***
- Examples:
  - ***Lists:*** $\mathbb{N} \lhd \lambda n. \{0, 1, \ldots, n-1\}$
  - ***Trees:*** $\mathsf{Tree}(A, P) \lhd \lambda t. \mathsf{Path}_{A,P}(t)$
- A ***container morphism*** $f \lhd g : (A \lhd P) \to (B \lhd Q)$ is given by

$$f : A \to B \qquad \text{and} \qquad g : \prod_{\{a:A\}} Q(f\, a) \to P\, a$$

- Containers and their morphisms form a category **Cont**

Capturing tree representations more abstractly: **cointerp. of conts.**

- Define the ***functor***[1] $\langle\!\langle - \rangle\!\rangle : \mathbf{Cont}^{\mathrm{op}} \to \mathbf{Type}$ as

$$\langle\!\langle A \lhd P \rangle\!\rangle \stackrel{\text{def}}{=} \prod_{a:A} P\, a \qquad \text{and} \qquad \langle\!\langle f \lhd g \rangle\!\rangle \stackrel{\text{def}}{=} \lambda \alpha.\, \lambda b.\, g\,(\alpha\,(f\,b))$$

where $f \lhd g : (B \lhd Q) \to (A \lhd P)$

[1] It arises from the ***cointerpretation of containers*** [A., Uustalu '14], given by $X \mapsto \prod_{a:A} (P\,a \times X)$

Capturing tree representations more abstractly: **cointerp. of conts.**

- Define the ***functor***[1] $\langle\!\langle - \rangle\!\rangle : \mathbf{Cont}^{\mathrm{op}} \to \mathbf{Type}$ as

$$\langle\!\langle A \lhd P \rangle\!\rangle \stackrel{\mathrm{def}}{=} \prod_{a:A} P\,a \qquad \text{and} \qquad \langle\!\langle f \lhd g \rangle\!\rangle \stackrel{\mathrm{def}}{=} \lambda\alpha.\,\lambda b.\,g\,(\alpha\,(f\,b))$$

where $f \lhd g : (B \lhd Q) \to (A \lhd P)$

- A tree representation of $F : \left(\prod_{a:A} P\,a\right) \to \left(\prod_{b:B} Q\,b\right)$ may be thus ***rewritten*** as:

$$\begin{tikzcd}
\langle\!\langle A \lhd P \rangle\!\rangle \arrow[rr, "F"] \arrow[dr, "\mathsf{c}_{A,P}"'] & & \langle\!\langle B \lhd Q \rangle\!\rangle \\
& \langle\!\langle \mathsf{Tree}(A,P) \lhd \lambda t.\,\mathsf{Path}_{A,P}(t) \rangle\!\rangle \arrow[ur, "\langle\!\langle \mathsf{t}_F \lhd \mathsf{e}_F \rangle\!\rangle"'] &
\end{tikzcd}$$

[1] It arises from the ***cointerpretation of containers*** [A., Uustalu '14], given by $X \mapsto \prod_{a:A} (P\,a \times X)$

Capturing tree representations more abstractly: **tree monad**

- The ***tree monad*** $(\mathcal{T}, \eta, \mu)$ on containers:

$$\mathcal{T}(A \lhd P) \stackrel{\text{def}}{=} \mathsf{Tree}(A, P) \lhd \lambda t.\, \mathsf{Path}_{A,P}(t),$$

$$\eta_{A \lhd P} \stackrel{\text{def}}{=} \big(\lambda a.\, \mathsf{node}(a, \lambda p.\, \mathsf{leaf})\big) \lhd \big(\lambda\{a\}.\, \lambda(\mathsf{step}(p, \mathsf{stop})).\, p\big)$$

$$\mu_{A \lhd P} \stackrel{\text{def}}{=} \cdots$$

- **More abstractly:** $\mathcal{T}(A \lhd P) \cong \mathbf{lfp}(X \lhd Y).\, \mathsf{Id}^{\mathsf{c}} +^{\mathsf{c}} (A \lhd P) \circ^{\mathsf{c}} (X \lhd Y)$

Capturing tree representations more abstractly: **tree monad**

- The ***tree monad*** $(\mathcal{T}, \eta, \mu)$ on containers:

$$\mathcal{T}(A \lhd P) \stackrel{\text{def}}{=} \mathsf{Tree}(A, P) \lhd \lambda t.\, \mathsf{Path}_{A,P}(t),$$

$$\eta_{A \lhd P} \stackrel{\text{def}}{=} \big(\lambda a.\, \mathsf{node}(a, \lambda p.\, \mathsf{leaf})\big) \lhd \big(\lambda\{a\}.\, \lambda(\mathsf{step}(p, \mathsf{stop})).\, p\big)$$

$$\mu_{A \lhd P} \stackrel{\text{def}}{=} \cdots$$

- **More abstractly:** $\mathcal{T}(A \lhd P) \cong \mathbf{lfp}(X \lhd Y).\, \mathsf{Id}^{c} +^{c} (A \lhd P) \circ^{c} (X \lhd Y)$

- A tree representation of F may be thus ***further rewritten*** as:

$$\begin{tikzcd}
\langle\!\langle A \lhd P \rangle\!\rangle \arrow[rr, "F"] \arrow[dr, bend right, "\mathsf{c}_{A,P}"'] & & \langle\!\langle B \lhd Q \rangle\!\rangle \\
& \langle\!\langle \mathcal{T}(A \lhd P) \rangle\!\rangle \arrow[ur, bend right, "\langle\!\langle \mathsf{t}_F \lhd \mathsf{e}_F \rangle\!\rangle"'] &
\end{tikzcd}$$

Capturing tree representations more abstractly: **comodule**

- The ***tree monad*** $(\mathcal{T}, \eta, \mu)$ on containers:

$$\mathcal{T}(A \lhd P) \stackrel{\text{def}}{=} \mathsf{Tree}(A, P) \lhd \lambda t.\, \mathsf{Path}_{A,P}(t),$$

$$\eta_{A \lhd P} \stackrel{\text{def}}{=} \big(\lambda a.\, \mathsf{node}(a, \lambda p.\, \mathsf{leaf})\big) \lhd \big(\lambda\{a\}.\, \lambda(\mathsf{step}(p, \mathsf{stop})).\, p\big)$$

$$\mu_{A \lhd P} \stackrel{\text{def}}{=} \cdots$$

- **More abstractly:** $\mathcal{T}(A \lhd P) \cong \mathbf{lfp}(X \lhd Y).\, \mathsf{Id}^{\mathsf{c}} +^{\mathsf{c}} (A \lhd P) \circ^{\mathsf{c}} (X \lhd Y)$

- A tree representation of F may be thus ***further rewritten*** as:

$$\begin{tikzcd}
\langle\!\langle A \lhd P \rangle\!\rangle \arrow[rr, "F"] \arrow[dr, bend right, "\mathsf{c}_{A \lhd P}"'] & & \langle\!\langle B \lhd Q \rangle\!\rangle \\
& \langle\!\langle \mathcal{T}(A \lhd P) \rangle\!\rangle \arrow[ur, bend right, "\langle\!\langle \mathsf{t}_F \lhd \mathsf{e}_F \rangle\!\rangle"'] &
\end{tikzcd}$$

a ***right $\mathcal{T}$-comodule***:

- $\mathsf{c} : \langle\!\langle - \rangle\!\rangle \to \langle\!\langle - \rangle\!\rangle \circ \mathcal{T}$ $\rightsquigarrow$
- unit & assoc. laws

Capturing tree representations more abstractly: **comodule reprs.**

- Given a *monad* (T, η, μ) on **Cont** and a *right T-comodule* $(\langle\!\langle - \rangle\!\rangle, \mathsf{c})$, a functional

  $$F : \left(\prod_{a:A} P\,a\right) \to \left(\prod_{b:B} Q\,b\right)$$

  is $(T, \langle\!\langle - \rangle\!\rangle, \mathsf{c})$***-representable*** if there exists a morphism in $\mathbf{Cont}_T$

  $$\mathsf{t}_F \lhd \mathsf{e}_F : (B \lhd Q) \to T(A \lhd P)$$

  such that

$$\begin{tikzcd}
\langle\!\langle A \lhd P \rangle\!\rangle \arrow[rr, "F"] \arrow[dr, bend right, "\mathsf{c}_{A \lhd P}"'] & & \langle\!\langle B \lhd Q \rangle\!\rangle \\
& \langle\!\langle T(A \lhd P) \rangle\!\rangle \arrow[ur, bend right, "\langle\!\langle \mathsf{t}_F \lhd \mathsf{e}_F \rangle\!\rangle"'] &
\end{tikzcd}$$

Capturing tree representations more abstractly: **comodule reprs.**

- Given a *monad* (T, η, μ) on **Cont** and a *right T-comodule* $(\langle\!\langle - \rangle\!\rangle, \mathsf{c})$, a functional

  $$F : (\prod_{a:A} P\,a) \to (\prod_{b:B} Q\,b)$$

  is $(T, \langle\!\langle - \rangle\!\rangle, \mathsf{c})$***-representable*** if there exists a morphism in $\mathbf{Cont}_T$

  $$\mathsf{t}_F \lhd \mathsf{e}_F : (B \lhd Q) \to T(A \lhd P)$$

  such that

  $$\begin{tikzcd}
\langle\!\langle A \lhd P \rangle\!\rangle \arrow[rr, "F"] \arrow[dr, bend right, "\mathsf{c}_{A \lhd P}"'] & & \langle\!\langle B \lhd Q \rangle\!\rangle \\
& \langle\!\langle T(A \lhd P) \rangle\!\rangle \arrow[ur, bend right, "\langle\!\langle \mathsf{t}_F \lhd \mathsf{e}_F \rangle\!\rangle"'] &
\end{tikzcd}$$

- **Thm:** Repr. functionals form a category. Full functor from reprs. to repr. funs.

What **other examples of representations** are out there?

$$
\begin{tikzcd}
\prod_{a:A} P\,a = \langle\!\langle A \lhd P \rangle\!\rangle \arrow[rr, "F"] \arrow[dr, bend right, "\mathsf{c}_{A \lhd P}"'] & & \langle\!\langle B \lhd Q \rangle\!\rangle = \prod_{b:B} Q\,b \\
& \langle\!\langle T(A \lhd P) \rangle\!\rangle \arrow[ur, bend right, "\langle\!\langle \mathsf{t}_F \lhd \mathsf{e}_F \rangle\!\rangle"'] &
\end{tikzcd}
$$

. . .   . . .

Continuous functionals   ***Fin. supported*** functionals   ***Functional*** functionals   ***Exceptional*** functionals   ***Constant*** functionals   ***Interactive*** functionals   ***Partial*** functionals   ***Instance reductions***   . . .

Functional functionals

- Consider:
  - the *identity monad* $T \stackrel{\text{def}}{=} \mathsf{Id} : \mathbf{Cont} \to \mathbf{Cont}$ (i.e., $T(A \lhd P) \stackrel{\text{def}}{=} A \lhd P$)
  - the *identity comodule* $\mathsf{c} \stackrel{\text{def}}{=} \mathsf{id} : A \lhd P \to A \lhd P$

Functional functionals

- Consider:
  - the *identity monad* $T \stackrel{\text{def}}{=} \mathsf{Id} : \mathbf{Cont} \to \mathbf{Cont}$ (i.e., $T(A \lhd P) \stackrel{\text{def}}{=} A \lhd P$)
  - the *identity comodule* $\mathsf{c} \stackrel{\text{def}}{=} \mathsf{id} : A \lhd P \to A \lhd P$
- A representation of $F : \left(\prod_{a:A} P\,a\right) \to \left(\prod_{b:B} Q\,b\right)$ is given by maps

$$\mathsf{t}_F : B \to A \qquad \text{and} \qquad \mathsf{e}_F : \prod_{\{b:B\}} P\,(\mathsf{t}_F\,b) \to Q\,b$$

  such that $F\,h\,b = \mathsf{e}_F(h\,(\mathsf{t}_F\,b))$
- A ***functional functional*** F computes $F\,h\,b$ by a *single* query to h (on inst. $\mathsf{t}_F\,b$)

Functional (and exceptional) functionals

- Consider:
  - the *identity monad* $T \stackrel{\text{def}}{=} \mathsf{Id} : \mathbf{Cont} \longrightarrow \mathbf{Cont}$ (i.e., $T(A \lhd P) \stackrel{\text{def}}{=} A \lhd P$)
  - the *identity comodule* $\mathsf{c} \stackrel{\text{def}}{=} \mathsf{id} : A \lhd P \longrightarrow A \lhd P$
- A representation of $F : \left(\prod_{a:A} P\,a\right) \longrightarrow \left(\prod_{b:B} Q\,b\right)$ is given by maps

$$\mathsf{t}_F : B \longrightarrow A \qquad \text{and} \qquad \mathsf{e}_F : \prod_{\{b:B\}} P\,(\mathsf{t}_F\,b) \longrightarrow Q\,b$$

  such that $F\,h\,b = \mathsf{e}_F(h\,(\mathsf{t}_F\,b))$
- A ***functional functional*** F computes $F\,h\,b$ by a *single* query to h (on inst. $\mathsf{t}_F\,b$)
- **Note:** *Exc. monad* = ***exceptional functionals*** = *single query* or *default answer*

Finitely supported functionals

- Consider:
  - $T(A \lhd P) \stackrel{\text{def}}{=} (\mathcal{P}_f\, A) \lhd (\lambda S.\, \prod_{a:S} P\, a)$ ($\mathcal{P}_f\, A$ is fin. powerset/-type of A)
  - $\mathsf{c}_{A \lhd P}\, h\, S \stackrel{\text{def}}{=} h{\restriction_S}$

Finitely supported functionals

- Consider:
  - $T(A \lhd P) \stackrel{\text{def}}{=} (\mathcal{P}_f\, A) \lhd (\lambda S.\, \prod_{a:S} P\, a)$ $\quad$ ($\mathcal{P}_f\, A$ is fin. powerset/-type of A)
  - $\mathsf{c}_{A \lhd P}\, h\, S \stackrel{\text{def}}{=} h{\restriction}_S$
- A representation of $F : (\prod_{a:A} P\, a) \to (\prod_{b:B} Q\, b)$ is given by

$$\mathsf{t}_F : B \to \mathcal{P}_f\, A \qquad \text{and} \qquad \mathsf{e}_F : \prod_{\{b:B\}} (\prod_{a:\mathsf{t}_F\, b} P\, a) \to Q\, b$$

  such that $F\, h\, b = \mathsf{e}_F\, (h{\restriction}_{\mathsf{t}_F\, b})$

- A ***finitely supported fun.*** F computes $F\, h\, b$ by a *finitely many* queries to h
- **Note:** The set of queries depends *only* on b, akin to truth-table reductions

Instance reductions

- Consider predicates $\phi : A \to \textbf{Prop}$ and $\psi : B \to \textbf{Prop}$, and implication

$$(\forall x \in A.\, \phi\, x) \Rightarrow (\forall y \in B.\, \psi\, y)$$

- An ***instance reduction***: $\forall y{:}B.\, \exists x{:}A.\, \phi\, x \Rightarrow \psi\, y$ (e.g., Zorn's lemma implies AC)

Instance reductions

- Consider predicates $\phi : A \to \mathbf{Prop}$ and $\psi : B \to \mathbf{Prop}$, and implication

$$(\forall x \in A.\, \phi\, x) \Rightarrow (\forall y \in B.\, \psi\, y)$$

- An ***instance reduction***: $\forall y{:}B.\, \exists x{:}A.\, \phi\, x \Rightarrow \psi\, y$ (e.g., Zorn's lemma implies AC)

- Restrict containers to ***propositional containers*** $A \lhd\!| \,\phi$, where $\phi : A \to \mathbf{Prop}$

- Use the *inhabited powerset monad* $\mathcal{P}_+$ and the following comodule:

$$T(A \lhd\!| \,\phi) \stackrel{\text{def}}{=} (\mathcal{P}_+\, A) \lhd\!| \,(\lambda S.\, \exists x{:}S.\, \phi\, x) \qquad \mathsf{c}_{A \lhd| \phi}\, h\, S \stackrel{\text{def}}{=} \text{proof of } \exists x{:}S.\, \phi\, x$$

Instance reductions

- Consider predicates $\phi : A \to \mathbf{Prop}$ and $\psi : B \to \mathbf{Prop}$, and implication

$$(\forall x \in A.\, \phi\, x) \Rightarrow (\forall y \in B.\, \psi\, y)$$

- An ***instance reduction***: $\forall y{:}B.\, \exists x{:}A.\, \phi\, x \Rightarrow \psi\, y$ (e.g., Zorn's lemma implies AC)

- Restrict containers to ***propositional containers*** $A \lhd\!| \, \phi$, where $\phi : A \to \mathbf{Prop}$

- Use the *inhabited powerset monad* $\mathcal{P}_+$ and the following comodule:

$$T(A \lhd\!| \, \phi) \stackrel{\text{def}}{=} (\mathcal{P}_+\, A) \lhd\!| \, (\lambda S.\, \exists x{:}S.\, \phi\, x) \qquad \mathsf{c}_{A \lhd\!| \phi}\, h\, S \stackrel{\text{def}}{=} \text{proof of } \exists x{:}S.\, \phi\, x$$

- **Note:** *Identity monad* on prop. containers = ***functional instance reductions***

$$\exists (f : B \to A).\, \forall y{:}B.\, \phi\, (f\, y) \Rightarrow \psi\, y$$

More examples **from monads on shapes** (from monads on **Type**)

- Given some *existing monad* M on **Type**, we get a monad T on $\mathbf{Cont}(\mathbf{U})$ by
  - defining $T(A \lhd P) \stackrel{\text{def}}{=} (MA) \lhd P^{\star}$ (where $A : \mathbf{Type}$ and $P : A \to \mathbf{U}$)
  - when **U** carries a ***weak Mendler-style** M**-algebra structure*** given by $(-)^{\star}$

More examples **from monads on shapes** (from monads on **Type**)

- Given some *existing monad* M on **Type**, we get a monad T on $\mathbf{Cont}(\mathbf{U})$ by
  - defining $T(A \lhd P) \stackrel{\text{def}}{=} (MA) \lhd P^{\star}$ (where $A : \mathbf{Type}$ and $P : A \to \mathbf{U}$)
  - when $\mathbf{U}$ carries a ***weak Mendler-style*** M***-algebra structure*** given by $(-)^{\star}$
- Fun., exc., and fin. supp. functionals, and instance reductions are all examples

More examples **from monads on shapes** (from monads on **Type**)

- Given some *existing monad* M on **Type**, we get a monad T on $\mathbf{Cont}(\mathbf{U})$ by
  - defining $T(A \lhd P) \stackrel{\text{def}}{=} (MA) \lhd P^\star$ (where $A : \mathbf{Type}$ and $P : A \to \mathbf{U}$)
  - when **U** carries a ***weak Mendler-style** M**-algebra structure*** given by $(-)^\star$
- Fun., exc., and fin. supp. functionals, and instance reductions are all examples
- Take the *trivial monad* $MA \stackrel{\text{def}}{=} \mathbb{1}$,
  - $P^\star \star \stackrel{\text{def}}{=} \mathbb{1}$ captures ***constant functionals***
  - $P^\star \star \stackrel{\text{def}}{=} \prod_{a:A} P\, a$ captures ***self-representation*** of functionals

More examples **from monads on shapes** (from monads on **Type**)

- Given some *existing monad* M on **Type**, we get a monad T on **Cont**(**U**) by
  - defining $T(A \lhd P) \stackrel{\text{def}}{=} (MA) \lhd P^\star$ (where $A : \mathbf{Type}$ and $P : A \to \mathbf{U}$)
  - when **U** carries a ***weak Mendler-style*** M***-algebra structure*** given by $(-)^\star$
- Fun., exc., and fin. supp. functionals, and instance reductions are all examples
- Take the *trivial monad* $MA \stackrel{\text{def}}{=} \mathbb{1}$,
  - $P^\star \star \stackrel{\text{def}}{=} \mathbb{1}$ captures ***constant functionals***
  - $P^\star \star \stackrel{\text{def}}{=} \prod_{a:A} P\, a$ captures ***self-representation*** of functionals
- Take the *input-output monad* $M \stackrel{\text{def}}{=} \mathsf{IO}$,
  - $P^\star\, \mathtt{c} \stackrel{\text{def}}{=}$ "IO–traces through IO-comp. c" captures ***interactive functionals***
  - dom & cod of Fs change to $\langle\!\langle A \lhd P \rangle\!\rangle_R \stackrel{\text{def}}{=} \prod_{a:A} (R \Rightarrow P\, a \times R)$, where R is a ***runner***

Thank you! Questions?

$$
\begin{tikzcd}
\prod_{a:A} P\,a \;=\; \langle\!\langle A \lhd P \rangle\!\rangle \arrow[rr, "F"] \arrow[dr, bend right, "\mathsf{c}_{A \lhd P}"'] & & \langle\!\langle B \lhd Q \rangle\!\rangle \;=\; \prod_{b:B} Q\,b \\
& \langle\!\langle T(A \lhd P) \rangle\!\rangle \arrow[ur, bend right, "\langle\!\langle \mathsf{t}_F \lhd \mathsf{e}_F \rangle\!\rangle"'] &
\end{tikzcd}
$$

. . . . . .

Continuous functionals | ***Fin. supported*** functionals | ***Functional*** functionals | ***Exceptional*** functionals | ***Constant*** functionals | ***Interactive*** functionals | ***Partial*** functionals | ***Instance reductions*** | . . .