The Art of Saying “No” – How to Politely Eat Your Way Through an Infinite Meal

Dirk Pattinson, Imperial College London

(joint work with Neil Ghani and Peter Hancock)

IFIP WG 1.3

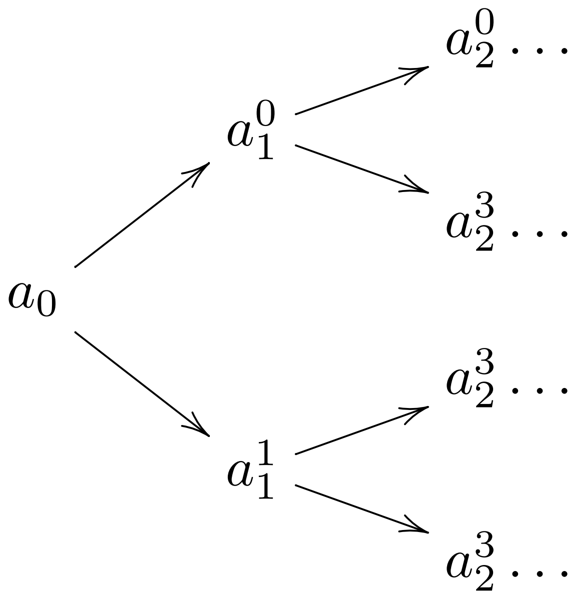
Udine September 2009

Infinite Objects are Coalgebras

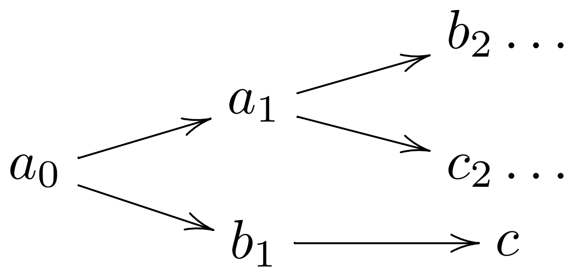
Infinite Streams over A: $\nu X.A \times X$

$$a_0 \longrightarrow a_1 \longrightarrow a_2 \longrightarrow \ldots$$

Infinite Binary Trees over A:
$\nu X.A \times X^2$



Signatures (variable branching):
$\nu X.A \times X^2 + B \times X + C$



Enter Topology . . .

Goal. *Algebraic* Treatment of *continuous* functions $\nu T \to \nu S$

- e.g. representatives of reals: $\{-1, 0, 1\}^\omega \rightsquigarrow [-1, 1]$
- clean (co)inductive definitions and proofs

Discrete Codomain. Continuous Functions $f : \nu X.TX \to B$

- output $b \in B$ after reading *finite amount* of information in $\nu X.TX$

Example. Infinite Streams, or coalgebraically $\nu X.A \times X \to B$

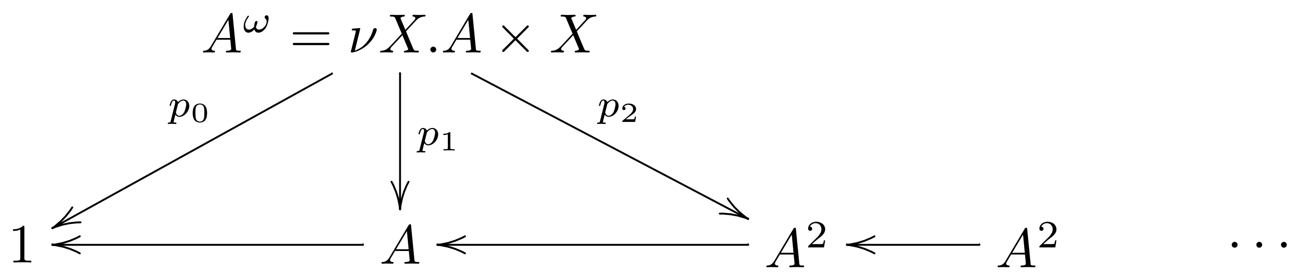
- $f(\alpha)$ depends on finite initial prefix of α

Conceptually. This is the Cantor topology on A^ω (with A discrete)

- generated by $\overline{\alpha} \cdot A^\omega$ where $\overline{\alpha} \in A^*$

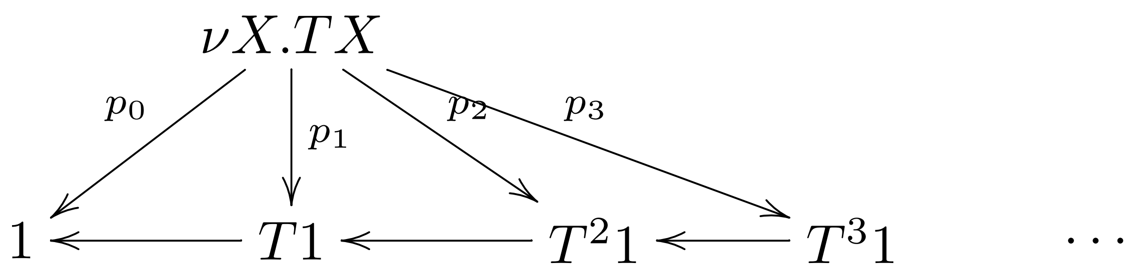
Coalgebraic View

Final Coalgebras arise as *infinite limits*: e.g. streams



Topology generated by $p_i^{-1}(o)$, $o \subseteq A^i$ open

Coalgebraic Generalisation. Suppose $\nu X.TX \xrightarrow{\sigma} T(\nu X.TX)$



where $p_{i+1} = Tp_i \circ \sigma$. Topology generated by $p_i^{-1}(o)$, "$o \subseteq T^i 1$ open"

Continuous Functions: The Case of Streams

Goal. Characterise continuous functions of type $A^\omega \to B$ with B discrete.

Continuity. (A and B discrete) $f : A^\omega \to B$ is continuous ...

- iff f locally constant.

$$(\forall(a_0, a_1, \dots) \in A^\omega)(\exists n \in \omega)\ f \text{ constant on } (a_0, a_1, \dots, a_n) \cdot A^\omega$$

- iff f is in the least class C closed under

$$\frac{f \text{ constant}}{C(f)} \qquad \frac{(\forall a \in A)C(f(a : \_))}{C(f)}$$

Proof ($\Leftarrow$) locally constant functions are so closed.

Proof ($\Rightarrow$) classical logic and dependent choice.

Representation of Continuous Stream Functions

Idea. Proofs of Continuity define the *least class* of functions

$$\frac{f \text{ constant}}{C(f)} \qquad \frac{(\forall a \in A) C(f(a : \_))}{C(f)}$$

and can be represented as an *inductive* data type:

$$R = \mu X.B + (A \to X) \cong B + (A \to R)$$

with two constructors: $\texttt{Ret} : B \to R$ and and $\texttt{Rd} : (A \to R) \to R$

from which a continuous function can be extracted:

$$\begin{array}{llll} \text{eat} : & \mu X.B + (A \to X) & \to A^\omega & \to B \\ \text{eat} & (\texttt{Ret} b) & (a : \alpha) & = b \\ \text{eat} & (\texttt{Rd} f) & (a : \alpha) & = \text{eat}(f\ a)\alpha \end{array}$$

Theorem. If $\to_c$ is continuous functions, then $\text{eat} : R \to (A^\omega \to_c B)$ is onto.

From Streams to Trees

Goal. Classify functions $\text{Tree}(A) \to_c B$ where $\text{Tree}(A) = \nu X.A \times X^2$

Idea. Let R denote the type of representatives with constructors $\texttt{Rd}$ and $\texttt{Ret}$.

$$\begin{array}{llll} \text{eat}: & R & \to \text{Tree}(A) & \to B \\ \text{eat} & (\texttt{Ret}\ b) & (a, l, r) & = b \\ \text{eat} & (\texttt{Rd}\ f) & (a, l, r) & = \text{eat}(f\ a)(l, r) \end{array}$$

Observation. $\text{eat}(f\ a) : \text{Tree}(A)^2 \to B$, so $f(a)$ represents $\text{Tree}(A)^2 \to B$

Mathematical Obfuscation. R_n represents $\text{Tree}(A)^n \to B$

$$\begin{array}{llll} \text{eat}_n: & R_n & \to \text{Tree}(A)^n & \to B \\ \text{eat}_n & (\texttt{Ret}\ b) & (t_1, \ldots, t_n) & = b \\ \text{eat}_n & (\texttt{Rd}_i\ f) & (t_1, \ldots, t_n) & = \text{eat}_{n+1}(f\ a_i)(t_1, \ldots, t_{i-1}, l, r, t_{i+1}, \ldots, t_n) \end{array}$$

where l, r are the left/right subtree of t_i. *Constructors.* $\texttt{Ret}, \texttt{Rd}_1, \ldots, \texttt{Rd}_n$

Escaping the Underworld of Indices

Desired (Inductive) Type with constructors $\mathtt{Ret}, \mathtt{Rd}_1, \ldots, \mathtt{Rd}_n$ as above.

$$R_n \cong B + \sum_{i \in n} (A \to R_{n+1})$$

Realisation Mapping.

$$\begin{array}{llll} \mathsf{eat}_n : & R_n & \to \mathsf{Tree}(A)^n & \to B \\ \mathsf{eat}_n & (\mathtt{Ret}\ b) & (t_1, \ldots, t_n) & = b \\ \mathsf{eat}_n & (\mathtt{Rd}_i\ f) & (t_1, \ldots, t_n) & = \mathsf{eat}_{n+1}(f\ a_i)(t_1, \ldots, t_{i-1}, l, r, t_{i+1}, \ldots, t_n) \end{array}$$

Taking Indices Seriously.

$$R(n) \cong B + \sum_{i \in n} (A \to R(n+1))$$

Observation. Now R has type $\mathsf{Set} \to \mathsf{Set}$ – and we want the *least* such

$$R = \mu F : \mathsf{Set} \to \mathsf{Set}. \Lambda I : \mathsf{Set}.\ B + I \times (A \to F(I+1))$$

Conceptual Digression

Streams. Represent $\mathsf{Stream}(A)^S \to B$ by $R(S)$ where

$$R(S) = \mu X.B + S \times (A \to X)$$

- each $R(S)$ is an initial algebra for a functor of type $\mathsf{Set} \to \mathsf{Set}$
- $\mathsf{eat}(S)$ defined by initiality of $R(S)$ – *separately* for all arities

Linearity: Family of Inductive Types

Trees. Represent $\mathsf{Tree}(A)^S \to_c B$ by $R(S)$ where

$$R = \mu F : \mathsf{Set} \to \mathsf{Set}.\Lambda S : \mathsf{Set}.B + S \times (A \to F(S+1))$$

- R is an initial algebra for a functor $(\mathsf{Set} \to \mathsf{Set}) \to (\mathsf{Set} \to \mathsf{Set})$
- eat is natural and defined by initiality of R – *simultaneously* for all arities

Nonlinearity: Inductive Family of Types

Infinite Objects of Container Type

Container Functors. (Abbot, Altenkirch, Ghani)

$$(S \lhd P)(X) = \sum_{s \in S} X^{P(s)}$$

- S : Set is a set of *shapes*, each of which stores data
- $P : S \to$ Set associates a set of *positions* to every shape

Continuous Functions of type $(\nu X.(S \lhd P)X)^I \to B$

$$R = \mu F : \mathsf{Set} \to \mathsf{Set}.\Lambda I : \mathsf{Set}.B + \sum_{i \in I} \prod_{s \in S} F(I + P(s))$$

Unfolding Isomorphisms.

$$R(I) \cong B + \sum_{i \in I} \prod_{s \in S} R(I + P(s))$$

Intuition.

- if not constant, select tree $(i \in I)$, extract root ($s \in S$), behead and continue

Discrete Codomains are Boring

Next Goal. Represent $A^\omega \to_c B^\omega$

Idea. $f : A^\omega \to B^\omega$ is continuous iff we have an *infinite* proof

$$(R)\frac{\forall a(C(f(a : \_)))}{C(f)} \qquad (W)\frac{C(f)}{C(\lambda\alpha.b : f(\alpha))}$$

where, on any branch in a proof, the right hand rule occurs infinitely often.

Induced Data Type. Wrap up finite occurrences of (R) using a μ

$$R \cong \nu X.\mu Y.B \times X + (A \to Y) \cong B \times R + (A \to R)$$

with constructors $\mathtt{Ret} : B \times R \to R$ and $\mathtt{Rd} : (A \to R) \to R$

Extracted Continuous Function.

$$\begin{array}{llll} \mathsf{eat} : & \nu X.\mu Y.B \times X + (A \to Y) & \to A^\omega & \to B^\omega \\ \mathsf{eat} & (\mathtt{Ret}\ (b, r)) & (a : \alpha) & = b : \mathsf{eat}\ r\ (a : \alpha) \\ \mathsf{eat} & (\mathtt{Rd}\ f) & (a : \alpha) & = \mathsf{eat}\ (f\ a)\ \alpha \end{array}$$

Alternative Computational Representation

We know. Continuous functions of type $A^w \to B$ are represented by

$$A^\omega \to_C B \rightsquigarrow R = \mu X.B + (A \to X)$$

Idea. Re-start the computation as soon as a digit has been produced

$$A^\omega \to_C B^\omega \rightsquigarrow \nu X.\mu Y.B \times X + (A \to Y)$$

with the same computational interpretation

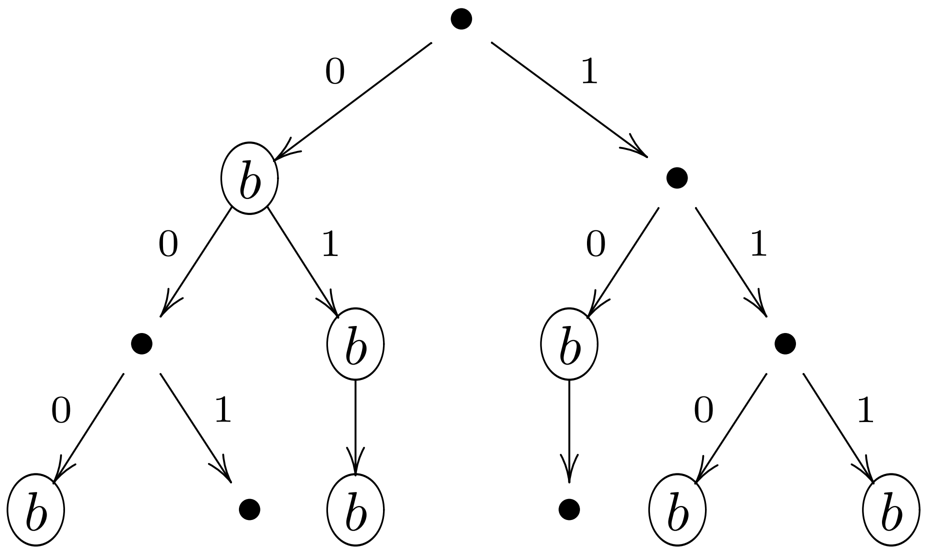
$$\begin{array}{llll} \text{eat}: & \nu X.\mu Y.B \times X + (A \to Y) & \to A^\omega & \to B^\omega \\ \text{eat} & (\texttt{Ret}\ (b, r)) & (a : \alpha) & = b : \text{eat}\ r\ (a : \alpha) \\ \text{eat} & (\texttt{Rd}\ f) & (a : \alpha) & = \text{eat}\ (f\ a)\ \alpha \end{array}$$

Note. Occurrence of $B \times X$ suggests that "codomain slots in"

Stream Functions are Trees

Observation. First-Order Functions $A^\omega \to B^\omega$ are *trees*

$$R = \nu X.\mu Y.B \times X + (A \to Y)$$



Initiality guarantees infinitely many labels on every path

General Codomain

More Ambitious Goal. Represent $A^\omega \to \nu X.(S \lhd P)X = \nu X.\sum_{s\in S} X^{P(s)}$

By Analogy.

$$R = \nu X.\mu Y.\sum_{s\in S} X^{P(s)} + (A \to Y) \cong \sum_{s\in S} R^{P(s)} + (A \to R)$$

with constructors $\mathtt{Ret}_s : (P(s) \to R) \to R$ and $\mathtt{Rd} : (A \to R) \to R$

Associated Functional.

$$\begin{array}{llll} \text{eat} : & \nu X.\mu Y.\sum_{s\in S} X^{P(s)} + (A \to Y) & \to A^\omega & \to \nu Z.(P \lhd S)Z \\ \text{eat} & (\mathtt{Ret}_s\ (r_i)) & (a : \alpha) & = (s, (\text{eat}\ r_i\ (a : \alpha))_{i\in P(s)}) \\ \text{eat} & (\mathtt{Rd}\ f) & (a : \alpha) & = \text{eat}\ (f\ a)\ \alpha \end{array}$$

Observation.

- *codomain* just “slots in”, more general *domains* by same recipie

Induction Meets Coinduction

Example. Continuous Stream Functions

$$f : A^\omega \to_c B^\omega$$

are represented by

$$\underbrace{\nu X.\, \overbrace{\mu Y \;\, B \times X + Y^A}^{T_A(B\times X)}}_{P_A(B)}$$

Lambek's Lemma.

$$P_A(B) = (\nu X)(\mu Y)B \times X + Y^A \cong (\mu Y)B \times P_A(B) + Y^A$$

Pleasant Mathematical Theory.

- supports both *inductive* and *coinductive* definitions and proofs.
- similar for other (co)domains

Inductive Maps Between Coinductive Types

Example. Composition: $P_B(C) \times P_A(B) \to P_A(C)$ where

$$T_A(B) = \mu X.B + (A \to X) \quad \text{and} \quad P_A(B) = \nu X.T_A(B \times X)$$

Operation on *representatives*

$$\begin{array}{ccc}
P_B(C) \times P_A(B) & \longrightarrow & P_A(C) \\
\downarrow & & \downarrow \\
(B^\omega \to_c C^\omega) \times (A^\omega \to_c B^\omega) & \longrightarrow & (A^\omega \to_c C^\omega)
\end{array}$$

As $P_A(B) \cong T_A(B \times P_A(B))$ is bi-inductive: composition

$$\gamma : S = T_B(C \times P_B C) \times T_A(B \times P_A B) \to T_A(C \times S)$$

is an *inductively defined* map between *coinductive* types

More on Composition

Inductive definition of composition

$$\begin{array}{llll}\gamma : S = T_B(C \times P_B C) \times T_A(B \times P_A B) \to T_A(C \times S) & & & \\ \langle\, \mathtt{Ret}\,\langle c, p_{bc}\rangle & , t_{ab} & \rangle & \mapsto \mathtt{Ret}\,\langle c, \mathsf{out}\, p_{bc}, t_{ab}\rangle \\ \langle\, \mathtt{Rd}\,\phi & , \mathtt{Ret}\,\langle b, p_{ab}\rangle & \rangle & \mapsto \gamma\langle \phi\, b, \mathsf{out}\, p_{ab}\rangle \\ \langle\, t_{bc} & , \mathtt{Rd}\,\psi & \rangle & \mapsto \mathtt{Rd}\,\lambda a.\gamma\langle t_{bc}, \psi\, a\rangle \end{array}$$

whose *coinductive* cousin (out : $\nu F \to F(\nu F)$)

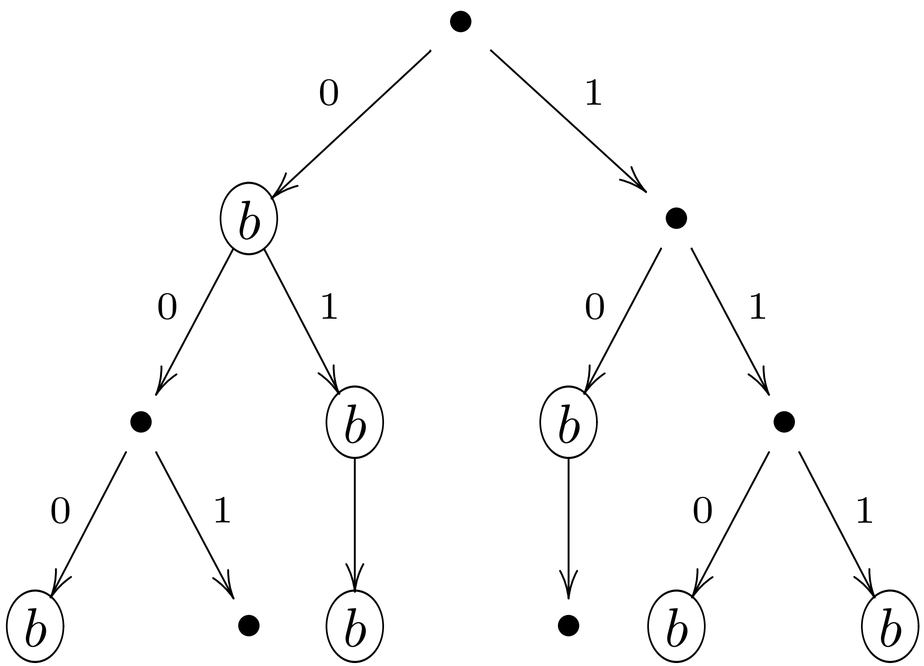
$$\begin{array}{lll}\chi : P_B(C) \times P_A(B) \to P_A(C) & & \\ \langle post, & pre\rangle & \mapsto (\mathsf{unfold}\,\gamma)\langle \mathsf{out}\, post, \mathsf{out}\, pre\rangle \end{array}$$

represents composition.

This is *output centered* – alternatives are possible.

Higher-Order Functions

Observation. First-Order Functions $A^\omega \to B^\omega$ are *trees*



Idea.

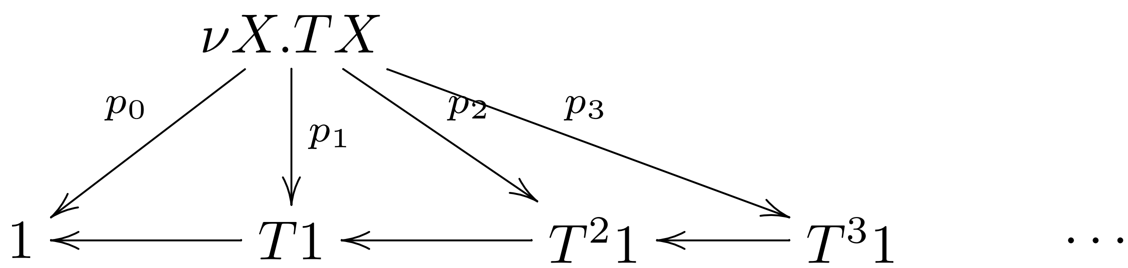
- represent higher-order functions as functions on trees
- *but:* domain doesn't fit into $\nu Z.(S \lhd P)Z = \nu Z.\sum_{s \in S} Z^{P(s)}$

Topological Excursion

Question. What's the natural topology on $A^\omega \to B^\omega$?

Topology on Representatives $R = \nu X.\mu Y.B \times X + (A \to Y)$.

- consider $TX = \mu Y.B \times X + (A \to Y)$ and $\sigma : R \to TR$
- topology given by the inverse limit



where $p_{i+1} = Tp_i \circ \sigma$. Topology generated by $p_i^{-1}(o)$, $o \subseteq T^i 1$ open

Induced Topology on $(A^\omega \to B^\omega)$ is compact-open:

- elements of $T^n 1$ are layers of A-branching trees with labels in B
- single trees define compact-open constraints

Container Magic

Summary so far.

- have representation of functions $\nu Z.(S \lhd P)Z \to X$
- want: representations of $\nu X.\mu Y.(B \times X) + (A \to Y) \to X$

Container Translation. Representations for free – if we solve

$$\mu Y.(B \times X) + (A \to Y) = (S \lhd P)X$$

Theorem. (Abbot/Alternkirch/Ghani) Containers are closed under μ, ν.

More precisely: for every n-ary container

$$C(X_1, \ldots, X_n) = \sum_{s \in S} X_1^{P_1(s)} \times \cdots \times X_n^{P_n(s)}$$

there is an $n-1$-ary container $D(X_1, \ldots, X_{n-1})$ that satisfies

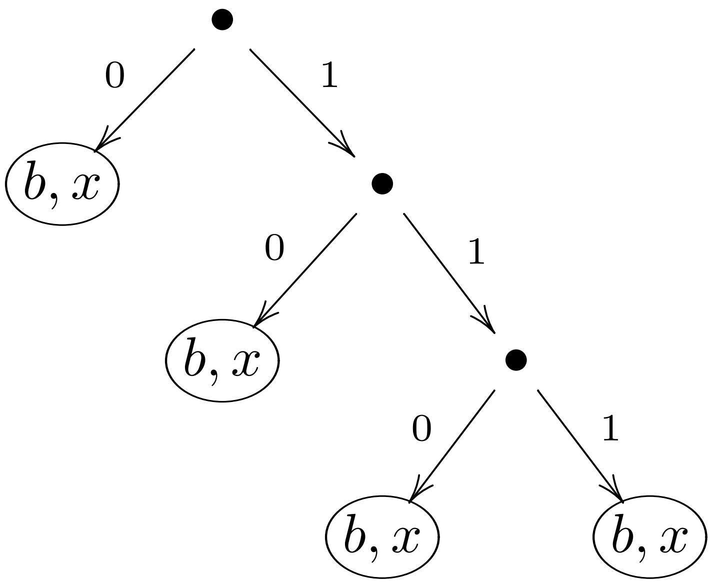
$$D(X_1, \ldots, X_{n-1}) = \mu X_n.C(X_1, \ldots, X_n)$$

Container Translation by Example

Wanted. Solutions of

$$\mu Y.B \times X + (A \to Y) \cong (S \lhd P)X = \sum_{s \in S} X^{P(s)}$$

Observation. We see *trees* with payload at the leaves.



Shapes.

$$S = \mu X.B + (A \to X)$$

Positions.

$P(s) = \{$ paths in S from root to leaves$\}$

Order-Two Example

Representatives. (Recall: $S = \mu X.B + (A \to X)$ and $P(s) =$ paths)

$$R = \nu F.\mu G.\Lambda I.\, C \times F(I) + I \times \prod_{s \in S} G(I + P(s))$$

Unfolding Isomorphisms. (Recall: $R(I)$ represents $(A^\omega \to B^\omega)^I \to C^\omega$)

$$R(I) \cong C \times R(I) + I \times \prod_{s \in S} R(I + P(s))$$

Induced Representation.

$$\begin{array}{llll}
\mathsf{eat}(I): & R & \to T^I & \to \nu Z.C \times Z \\
\mathsf{eat}(I) & (\mathtt{Ret}(c, r)) & (\phi) & = c : (\mathsf{eat}\ r\ \phi) \\
\mathsf{eat}(I) & (\mathtt{Rd}\,(i, f)) & (\phi) & = \mathsf{eat}\ (f(\mathsf{root}\,\phi(i)))\ [\phi, \mathsf{debris}(\phi(i))]
\end{array}$$

Notation. For $t = (r, d) \in T = \nu X.(S \lhd P)X \cong \sum_{s \in S} T^{P(s)}$

$$\mathsf{root}(r, d) = r \qquad \text{and} \qquad \mathsf{debris}(r, d) = d$$

Conclusions

Tree Eating.

- linear structures (streams) $\leadsto$ family of inductive types
- nonlinear structures (trees) $\leadsto$ inductive families of types
- in both cases: sound and complete representation of continuous functions

Higher Order Functions.

- reducible to tree case – but with coding
- possibly very inefficient in practice – try out

Open Questions.

- more combinators (e.g. buffering, currying)
- concrete case studies – in particular integration
- complexity of (higher order) stream functions?