Containers MSP101

Fredrik Nordvall Forsberg

19th November 2020

Why?

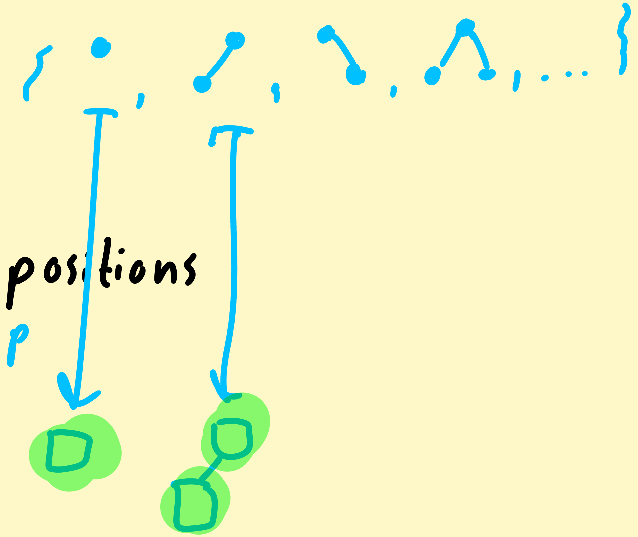
Tuples, lists, binary trees, finitely branching trees, streams, infinite trees, ...

- Generic programs?
- Generic theorems?
- A core theory we can understand.

What is a container?

A container $S \triangleleft P$ is given by

- a set S of shapes

  e.g. $S = \{\emptyset, \bullet, \bullet\!-\!\bullet, \bullet\!-\!\bullet\!-\!\bullet, \ldots\}$ or $S = \{$ 

- for each shape $s \in S$, a set $P(s)$ of positions

  e.g. $P(\emptyset) = \emptyset$ $\quad P(\bullet\!-\!\bullet) = \square\ \square$

  $P(\bullet) = \square$

Examples

Tuples $S = \{*\}$ $P(*) = \{\text{left}, \text{right}\}$

Lists $S = \mathbb{N}$ $P(n) = \{1, \ldots, n\}$

Binary trees $S =$ "shapes of trees" $P(s) = \{$"nodes in s"$\}$

What can you do with it?

$S = \mathbb{N}$

$P(n) = \{1, \dots, n\}$

Given a payload X, eg $X = \mathbb{N}$

choose a shape $s \in S$, eg $s =$ 

for every position $p \in P(s)$,

fill it with a payload from X.

The extension of a container

For a container $S \triangleright P$, we define $[\![S \triangleright P]\!] : \mathrm{Set} \to \mathrm{Set}$ by

$$[\![S \triangleright P]\!]\, X = \Sigma s \in S.\, (P(s) \to X)$$

$\Sigma x \in A.\, B(x) = \{(a, b) \mid a \in A,\ b \in B(a)\}$

This extends to a functor $\mathrm{Set} \to \mathrm{Set}$:

$$P(s) \xrightarrow{h} X \xrightarrow{f} Y$$

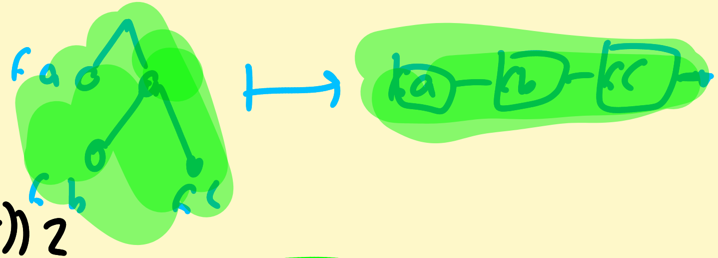
$$[\![S \triangleright P]\!](f : X \to Y) : \Sigma s \in S.\, (P(s) \to X) \to \Sigma s' \in S.\, (P(s') \to Y)$$

$$(s, h) \longmapsto (s, f \circ h)$$

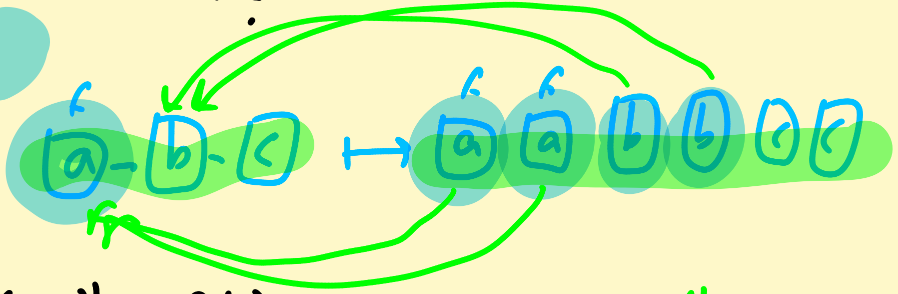
Container morphisms

How can we represent natural transformations $[\![S \triangleright P]\!] \xrightarrow{\cdot} [\![S' \triangleright P']\!]$?

$\text{flatten}_X : \text{Tree}\ X \to \text{List}\ X$



$f : \underline{S} \to \underline{S'} \quad g : \forall s.\ P(s) \to P'(f(s))$?



$\text{double}_X : \text{List}\ X \to \text{List}\ X$

$f : s \to s' \quad g : \forall s.\ P'(f(s)) \to P(s)$

"where did the new position come from?"

The extension of a container morphism

$f : S \to S'$ $\quad$ $g : \prod_{s \in S}. \left(P'(f(s)) \to P(s)\right)$

$$[\![f \triangleright g]\!]_X : \overset{\Sigma_{s\in S}.(P(s)\to X)}{\overset{\shortparallel}{[\![S \triangleright P]\!]X}} \to \overset{\Sigma_{s'\in S'}.(P'(s')\to X)}{\overset{\shortparallel}{[\![S' \triangleright P']\!]X}}$$

$$(s, h) \longmapsto (f(s), h \circ g_s)$$

[Note: if P and P' are constant, then this is exactly a lens $\binom{S}{P} \to \binom{S'}{P'}$!]

$$\begin{tikzcd}
P'(f(s)) \arrow[dr, "g_s"'] & & X \\
& P(s) \arrow[ur, "h"'] &
\end{tikzcd}$$

<u>Thm</u> There is a category Cont and a functor $[\![-]\!] : \mathrm{Cont} \to (\mathrm{Set} \to \mathrm{Set})$.

functor cat

Full and faithfulness of $[\![\cdot]\!]$

Thm $[\![-]\!]$ is full and faithful, i.e. $\mathrm{Hom}_{\mathrm{Set}\to\mathrm{Set}}([\![S \triangleright P]\!], [\![S' \triangleright P']\!]) \cong \mathrm{Hom}_{\mathrm{Cont}}(S \triangleright P, S' \triangleright P')$.

Proof:

$$\mathrm{Hom}_{\mathrm{Set}\to\mathrm{Set}}(\Sigma s \in S.\ P(s) \to -,\ [\![S' \triangleright P']\!])$$
$$\cong \Pi s \in S.\ \mathrm{Hom}(P(s) \to -,\ [\![S' \triangleright P']\!])$$
$$\cong \Pi s \in S.\ [\![S' \triangleright P']\!](P(s))$$
$$= \Pi s \in S.\ \Sigma s' \in S'.\ (P'(s') \to P(s))$$
$$\cong \Sigma f : S \to S'.\ \Pi s \in S.\ P'(f(s)) \to P(s)$$

Yoneda Lemma

Let $F : \mathbb{C} \to \mathrm{Set}$ and $A \in \mathbb{C}$.

Then $\mathrm{Hom}_{\mathbb{C}\to\mathrm{Set}}(A \to -, F) \cong F(A)$.

$(A \to A) \to FA$

$id \mapsto$

Time for amazement

The collection of nat. trans. $[S \triangleright P] \dot\to [S' \triangleright P']$ is a priori very large — each element quantifies over all sets!

The theorem says that all these large things are in fact represented by two small functions.

Concretely:

- There are exactly ~~two~~ four nat. trans. $\mathrm{Tuple}\, X \longrightarrow \mathrm{Tuple}\, X$

  id, swap, $\mathrm{const}_{\mathrm{first}}$, $\mathrm{const}_{\mathrm{snd}}$

- Every nat. trans. $\mathrm{List}^{2^2} X \to \mathrm{List}\, X$ is given by

  a length transformer $f : \mathbb{N} \longrightarrow \mathbb{N}$ $n \mapsto 2n$ $0 \mapsto 0$, $n \mapsto n-1$

  a position rearranger $g : \prod n \in \mathbb{N}.\ (\underline{\{1,\dots,f(n)\}} \longrightarrow \underline{\{1,\dots,n\}})$

Closure I: Id and constant functors

$$\underline{\mathrm{Id}}(X) = X \underset{x \mapsto (*,\, x)}{\cong} \sum_{s \in \underline{1}} X \cong \sum_{s \in 1} (1 \to X) = \underline{[\![1 \rhd 1]\!]}(X)$$

$$K(X) = A \cong \sum_{s \in A} (0 \to X) = [\![A \rhd 0]\!](X)$$

Closure II: Products and coproducts

$$
\begin{aligned}
(\llbracket S \rhd P \rrbracket \times \llbracket S' \rhd P' \rrbracket)\, X &= \textstyle\sum_{s \in S}.\, (P(s) \to X) \times \sum_{s' \in S'}.\, (P'(s') \to X) \\
&\cong \textstyle\sum_{(s,s') \in S \times S'}.\, \big((P(s) \to X) \times (P'(s') \to X) \big) \\
&\cong \textstyle\sum_{(s,s') \in S \times S'}.\, \big((P(s) + P'(s')) \to X \big) \\
&= \llbracket S \times S' \rhd P(-) + P'(-) \rrbracket
\end{aligned}
$$

$$
\begin{aligned}
(\llbracket S \rhd P \rrbracket + \llbracket S' \rhd P' \rrbracket)\, X &= \textstyle\sum_{s \in S}.\, (\overline{P(s)} \to X) + \sum_{s' \in S'}.\, (\overline{P'(s')} \to X) \\
&\cong \textstyle\sum_{x \in S + S'}.\, \begin{cases} P(s) \to X & \text{if } x = \mathrm{inl}\, s \\ P'(s') \to X & \text{if } x = \mathrm{inr}\, s' \end{cases} \\
&\cong \textstyle\sum_{x \in S + S'}.\, \begin{pmatrix} P(s) \text{ if } x = \mathrm{inl}\, s \\ P'(s') \text{ if } x = \mathrm{inr}\, s' \end{pmatrix} \to X \\
&= \llbracket S + S' \rhd \underline{\begin{pmatrix} P(s) \text{ if } x = \mathrm{inl}\, s \\ P'(s') \text{ if } x = \mathrm{inr}\, s' \end{pmatrix}} \rrbracket
\end{aligned}
$$

Note: even if P, P' are constants, the new positions are def. dependent

Closure III: Least and greatest fixed points

Given $S \triangleright P$, want (least) X s.t. $[\![S \triangleright P]\!](X) \cong X$.

Is this a container? Obviously not, because the question does not typecheck!

Generalise to n-ary containers:

S set

$P_1 : S \to \text{set}$

$P_2 : S \to \text{se}$

$\vdots$

$P_n : S \to \text{set}$

$[\![S \triangleright \vec{P}]\!] : \text{set}^n \to \text{set}$

$[\![S \triangleright \vec{P}]\!](X_1, \dots, X_n) = \Sigma s \in S. \, (P_1(s) \to X_1) \times \dots \times (P_n(s) \to X_n)$

Thm For $(n+1)$-ary container $S \triangleright \vec{P}$,

$\mu X_{n+1}. \, [\![S \triangleright \vec{P}]\!](X_1, \dots, X_{n+1})$

also ν

is an n-ary container

$S' =$ "trees of $[\![S \triangleright P]\!]$"

$P' =$ "paths into the tree"

Generalisations

- Set $\rightsquigarrow$ $\underbrace{\text{locally cartesian closed category with extensive coproducts}}$

  $\Rightarrow$ has internal language with Σ, Π, so can do what we did internally

- Containers $\rightsquigarrow$ indexed containers

  Represents functors $(I \to Set) \to (O \to Set)$

  Indexed data types such as vectors, relations, etc

Summary

- Container $S \triangleleft P$ given by set S and $P : S \to Set$
- Category Cont with full and faithful $[\![-]\!] : Cont \to (Set \to Set)$
- Closed under most things you can imagine

References

Abbott: Categories of containers

Abbott, Altenkirch, Ghani: Containers: constructing strictly positive types

Gambino, Hyland: Wellfounded trees and dependent polynomial functors

J. Kock: Polynomial functors and trees

Altenkirch, Ghani, Hancock, McBride, Morris: Indexed containers

$$\begin{pmatrix} A \\ B \end{pmatrix} \in \text{Obj Lens} \qquad\qquad \begin{pmatrix} s : A \\ p(s) : B \end{pmatrix} \text{Obj Cont}$$

$$\begin{array}{ccc} \begin{pmatrix} A \\ B \end{pmatrix} & & \begin{pmatrix} s : A \\ p(s) : B \end{pmatrix} \\ \downarrow & & \downarrow \\ \begin{pmatrix} A' \\ B \end{pmatrix} & & \begin{pmatrix} A' \\ B' \end{pmatrix} \end{array}$$

get: $A \to A'$

put: $A \times B' \to B$

$$\begin{pmatrix} A \\ B \end{pmatrix} + \begin{pmatrix} A' \\ B' \end{pmatrix}$$

$$[\![A \triangleright B]\!] \qquad X \mapsto A \times (B \to X)$$

$$[\![S \triangleright P]\!]\, 1 = \sum_{s \in S.} \overbrace{(P(s) \to 1)}^{1} \cong S$$