COMODELS OF AN ALGEBRAIC THEORY

1) Equational algebraic theories

A signature is a set Σ of operations σ, each assigned an arity $|\sigma| \in \begin{cases} \mathbb{N} & \text{– finitary} \\ \mathrm{Set} & \text{– infinitary} \end{cases}$.

Eg: the signature for groups Σ_{Gp} is $\{\bullet, e, (-)^{-1}\}$ w/ arities 2, 0, 1

Given a sign. Σ and a set A, can recursively define set $\Sigma(A)$ of Σ-terms with free variables in A:

Eg: $\Sigma_{Gp}(\{x, y, z\}) \ni (x \cdot y) \cdot z^{-1}, \ (y \cdot e^{-1}) \cdot (x \cdot x), \ldots$

An alg. theory $\mathbb{T}$ is a signature Σ, plus a set of equations: each eqn. is $s = t$ for some A and $s, t \in \Sigma(A)$.

Eg: $\mathbb{T}_{Gp}$ involves equations $(x \cdot y) \cdot z = x \cdot (y \cdot z), \ x \cdot x^{-1} = e, \ldots$

If $\mathbb{T}$ is an alg theory, A a set, write $T(A)$ for set of $\mathbb{T}$-terms w/ variables in A: its the quotient of $\Sigma(A)$ by $\mathbb{T}$-derivable equality.

Eg: $T_{Gp}(A)$ is the free group with gen. set A.

In computer science, we use algebraic theories $\mathbb{T}$ to encode

<u>notions of computation</u> (Moggi 1991, Plotkin - Power 2001, 2, 3 ...): $T(A)$ is the set of programs with computation in $\mathbb{T}$ returning values in A.

<u>Eg</u>: let V be a set. The theory of <u>V-ary input</u> $\mathbb{T}_{In}$ has one V-ary operation read, and no equations. See elements of $T_{In}(A)$ as programs returning vals in A as follows:

$$a \in A \subseteq T(A) \rightsquigarrow \text{return } a$$
$$\mathrm{read}(\lambda v.\ t_v) \rightsquigarrow \text{let } v \text{ be read() in } t_v$$

<u>Eg</u>: when $V = \mathbb{N}$, we have a program in $T_{In}(\mathbb{N})$:

$\mathrm{read}(\lambda v.\ \mathrm{read}(\lambda w.\ v+w)) \rightsquigarrow$ let v be read() in
let w be read() in
return $v+w$.

$$\left(\begin{array}{l} \text{do} \quad v \leftarrow \text{read} \\ \qquad w \leftarrow \text{read} \\ \qquad \text{return } v+w \end{array} \right)$$

<u>Eg</u>: Let V be a set. Theory of a <u>V-valued stack</u> (Goncharov 2001) has:

- unary operations push_v $(v \in V)$
- $V + \{\perp\}$-ary operation pop

interpreted as:

$\mathrm{push}_v(x) \rightsquigarrow$ push v; x

$\mathrm{pop}(\lambda v. x_v, y) \rightsquigarrow$ try

let v be pop() in x_v

catch y

Equations are:

(for each $v \in V$) $\mathrm{push}_v(\mathrm{pop}(\vec{x}, y)) = x_v \qquad \mathrm{pop}(\lambda v.\ \mathrm{push}_v(x), x) = x$

$$\mathrm{pop}(\vec{x}, \mathrm{pop}(\vec{y}, z)) = \mathrm{pop}(\vec{x}, z).$$

2) COMODELS OF ALG. THEORIES

There's a standard notion of model of an alg theory $\mathbb{T}$: for example a $\mathbb{T}_{Gp}$-model is a group.

More generally, can look at $\mathbb{T}$-models in any caty $\mathcal{C}$ with products. In particular, can take $\mathbb{T}$-models in Set^{op}: call these comodels. These involve:

- a set S;
- a cointerpretation $[\![\sigma]\!] : S \to |\sigma| \times S \quad \forall \sigma \in \Sigma$;
- ... inducing derived cointerps. $[\![t]\!] : S \to A \times S \quad \forall\, t \in \Sigma(A)$;
- which satisfy $[\![s]\!] = [\![t]\!] \quad \forall$ eqns $s = t$ of $\mathbb{T}$.

Eg: A $\mathbb{T}_{Gp}$-comodel involves $[\![e]\!] : S \to \emptyset$; so only $\mathbb{T}_{Gp}$-comodel is $\emptyset$.

Eg: A $\mathbb{T}_{In}$-comodel is a set S (of "states") t/w a cointerpretation $[\![\text{read}]\!] : S \longrightarrow V \times S$, with no further conditions. Think that $[\![\text{read}]\!]$ takes a state s, and yields a new input value from V, and a next state s'.

Eg: A $\mathbb{T}_{\text{Stack}}$-comodel involves a set S of states, and

$$[\![\text{push}_v]\!] : S \longrightarrow S \qquad \forall v \in V$$
$$[\![\text{pop}]\!] : S \longrightarrow (V + \{\perp\}) \times S$$

st:

1) $[\![\text{pop}]\!]([\![\text{push}_v]\!](s)) = (v, s)$
2) $[\![\text{pop}]\!](s) = (v, s') \implies [\![\text{push}_v]\!](s') = s$
3) $[\![\text{pop}]\!](s) = (\perp, s') \implies s' = s.$

=

In general (Power-Shkaravska 2004, ...) if $\mathbb{T}$-terms are programs interacting with an environment, then $\mathbb{T}$-comodels are instances of that environment. Moreover, the cointerpretation of a program $t \in T(A)$ in a comodel $(S, [\![-]\!])$ is a function $[\![t]\!] : S \longrightarrow A \times S$ <u>running</u> the computation t using state machine $(S, [\![-]\!])$, starting from some state $s \in S$, and returning a value in A and a final state $s' \in S$.

Eg: If have the program $p = \text{read}(\lambda v.\ \text{read}(\lambda w.\ v + w)) \in T_{\text{In}}(\mathbb{N})$, for the comodel with $S = \{a, b, c\}$ and

$$[\![\text{read}]\!] : \begin{aligned} a &\longmapsto (12, a) \\ b &\longmapsto (7, c) \\ c &\longmapsto (4, b) \end{aligned}$$

We have $[\![p]\!] : S \longrightarrow \mathbb{N} \times S$

$$\begin{aligned} a &\longmapsto (24, a) \\ b &\longmapsto (11, b) \\ c &\longmapsto (11, c) \end{aligned}$$

3) BEHAVIOUR CATEGORIES

Comodels of any algebraic thy $\mathbb{T}$, with their homomorphisms, form a caty $\mathrm{Comod}(\mathbb{T})$.

THEOREM (G., 2020) For any alg thy $\mathbb{T}$, $\mathrm{Comod}(\mathbb{T})$ is a presheaf category $[\mathbb{B}_{\mathbb{T}}, \mathrm{Set}]$.

We call $\mathbb{B}_{\mathbb{T}}$ the behaviour caty of $\mathbb{T}$. — we can calculate it very explicitly!

Defn Behaviour caty $\mathbb{B}_{\mathbb{T}}$ of an alg thy $\mathbb{T}$ has:

- objects are admissible behaviours: families of fns
$$(\beta_A : T(A) \longrightarrow A)_{A \in \mathrm{Set}}$$
st:
$$\beta(a) = a \qquad \forall\, a \in A \subseteq TA$$
$$\beta(t(\lambda a.\, u_a)) = \beta(t(\lambda a.\, u_{\beta(t)})) \quad \cdots \quad \beta(t \gg= u) = \beta(t \gg u_{\beta(t)}).$$

- maps $\beta \to \beta'$ are $\{ m \in T(1) \mid \beta' = \beta(m(-)) \} / \sim_\beta$

where $\sim_\beta$ smallest equiv relation st $(t >>= m) \sim_\beta (t >> m_{\beta(t)})$

Eg: For theory $\mathbb{T}_{ln}$, behaviour caty looks like:

- objects are infinite words $W \in V^{\mathbb{N}}$
- maps $W \longrightarrow W'$ is $n \in \mathbb{N}$ st $\partial^n W = W'$.

Eg: For theory $\mathbb{T}_{stack}$, behaviour caty looks like: -

- objects: finite or infinite words $W \in V^{\leq \omega}$
- maps: * $!$ map $W \longrightarrow W'$ if both are finite;
  * no maps $W \longrightarrow W'$ if only one is finite;
  * $W \rightarrow W'$, where W, W' infinite, is some $i \in \mathbb{Z}$ st for some $n \in \mathbb{N}$, $\partial^n W = \partial^{n+i} W'$