

# *Fixpoints, Categories, and (Co)Algebraic Modeling*

Peter Padawitz  
TU Dortmund, Germany  
July 11, 2021

(actual version: <https://fdit-www.cs.tu-dortmund.de/~peter/DialgSlides.pdf>)

# Contents

The central chapters have boldface titles.

1	<b>The Tai Chi of algebraic modeling</b>	9
2	Preliminaries	10
	Products	13
	Sums	19
	Miscellanea	26
3	Relations, ordered sets, and fixpoints	38
4	Categories	59
5	Functors and natural transformations	74
6	Limits and colimits	86
	Limits	87
	Colimits	96
7	<b>Sorted sets and types</b>	105
	Type models	110
	Lifting of sorted relations	114

8 Signatures	118
$\Sigma$ -arrows	119
Derived $\Sigma$ -arrows	120
Sample constructive signatures	123
Sample destructive signatures	128
9 $\Sigma$ -algebras	138
Sample algebras	148
Products, sums, invariants and quotients of $\Sigma$ -algebras	171
$\Sigma$ -terms and -coterms	183
Sample terms and coterms	187
Term and coterminous algebras	193
Term folding (“denotational semantics”)	195
Sample initial algebras	203
Context-free grammars with base languages	207
State unfolding (“operational semantics”)	214
Sample final algebras	229
$\Sigma$ -flowcharts	244
$\Sigma$ -formulas: Syntax	256
Derived $\Sigma$ -formulas	262
$\Sigma$ -formulas: Semantics	269
Automata for satisfiability	285

Institutions	293
<b>10 Sequent logic</b>	<b>295</b>
Continuity of step functions and consequences	306
Deduction	318
Rule applicability	321
Resolution and narrowing	322
<b>11 Fixpoint induction and coinduction</b>	<b>327</b>
Incremental induction	332
Incremental coinduction	334
Duality of (co)resolution and (co)induction	342
<b>12 <math>F</math>-algebras and <math>F</math>-coalgebras</b>	<b>344</b>
Invariants and congruences on $F$ -(co)algebras	355
Complete categories and continuous functors	358
Categorical construction of initial $F$ -algebras and final $F$ -coalgebras	362
<b>13 <math>\Sigma</math>-functors</b>	<b>375</b>
Functors for constructive signatures	375
Functors for destructive signatures	379
Final models of non-polynomial signatures	384
Recursive functions	392
Bisimulation modulo constructors	408

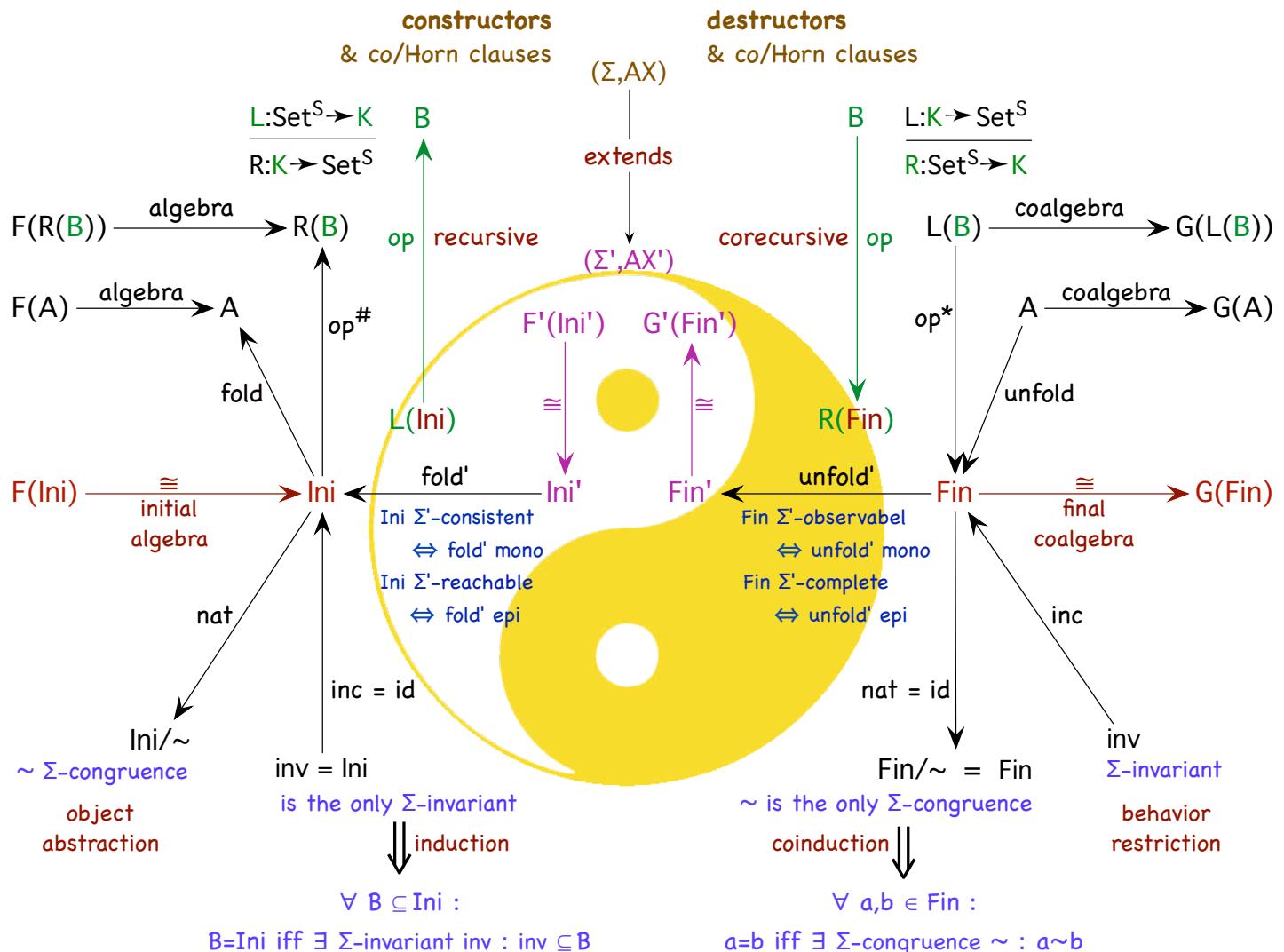
Sample inductively defined functions	410
Sample coinductively defined functions	434
Sample biinductively defined functions	453
Biinductive definition of regular operators	455
From constructors to destructors	457
From destructors to constructors	461
Continuous algebras	465
<b>14 Iterative equations</b>	479
Term equations	480
Flowchart equations	502
Word acceptors	509
Tree acceptors	516
Iterative equations of a CFG	525
<b>15 Algebraic induction and coinduction</b>	532
Invariants and algebraic induction	532
Algebraic induction as fixpoint induction	538
Invariants are monotone	539
Congruences and algebraic coinduction	543
Algebraic coinduction as fixpoint coinduction	549
Quotients are monotone	561

16 Categorical $\Sigma$ -algebra constructions	565
Initial models of constructive polynomial signatures	565
Final models of destructive polynomial signatures	569
Bounded functors	572
17 Adjunctions	581
Sample adjunctions	585
1. Identity functor	585
2. Monoid functor	585
3. Sequence functor	587
4. Behavior functor	589
5. Weighted-set functors	591
6. Box and diamond functors	593
7. Strongly connected graph components	595
8. Reader and writer	597
9. Product	599
10. Coproduct	601
11. Term functors	603
12. Varieties	608
13. Coterm functors	622
14. Covarieties	625
15. Base algebra extensions	644

18 Stream calculus	646
19 Conservative extensions	665
Constructor extensions	665
Destructor extensions	668
20 Abstraction	671
Abstraction with a least congruence	674
Abstraction with a greatest congruence	677
21 Restriction	680
Restriction with a greatest invariant	683
Restriction with a least invariant	687
22 $\lambda$ -bialgebras	691
23 Monads and comonads	696
Sample monads	698
The sequence monad	703
Term monads	704
Sample comonads	711
The behavior comonad	713
Coterm comonads	714
24 Recursive functions defined by adjunctions or distributive laws	721

25 External examples	730
Queues	730
Arithmetic expressions	732
CCS	741
26 Bibliography	758

# The Tai Chi of algebraic modeling



## Preliminaries

$\textcolor{red}{1} =_{\text{def}} \{()\}$ ,  $\textcolor{red}{2} =_{\text{def}} \{0, 1\}$ ,  $\mathbb{N}_+ =_{\text{def}} \mathbb{N} \setminus \{0\}$ ,  $[0] = \emptyset$  and for all  $n > 0$ ,  $[\textcolor{red}{n}] =_{\text{def}} \{1, \dots, n\}$ .

Let  $A, B$  be sets and  $I$  be a nonempty set.

Both  $\textcolor{red}{A} \rightarrow \textcolor{red}{B}$  and  $\textcolor{red}{B}^{\textcolor{red}{A}}$  denote the set of functions from  $A$  to  $B$

$|A|$  denotes the **cardinality** of  $A$ .  $|A| < \omega$  means that  $A$  is finite.

$\textcolor{teal}{id}_A : A \rightarrow \textcolor{blue}{A}$  denotes the **identity** on  $A$  that maps every element of  $A$  to itself.

$\Delta_A^I = \{(a)_{i \in I} \mid a \in A\}$  denotes the  **$I$ -dimensional diagonal** of  $A$ .

In particular,  $\Delta_A =_{\text{def}} \Delta_A^{[2]}$ .

Let  $f, g : A \rightarrow B$  and  $C \subseteq A$ .

$A$  is the **domain** of  $f$ .  $B$  is the **range** of  $f$ .

$\textcolor{red}{graph}(f) =_{\text{def}} \{(a, b) \in A \times B \mid f(a) = b\}$  is called the **graph** of  $f$ .

$\textcolor{red}{img}(f) =_{\text{def}} \{f(a) \mid a \in A\}$  is called the **image** of  $f$ .

$\ker(f) =_{def} \{(a, a') \in A^2 \mid f(a) = f(a')\}$  is called the **kernel** of  $f$ .

$f$  is **surjective** if  $\text{img}(f) = B$ .  $f$  is **injective** if  $\ker(f) = \Delta_A$ . If both conditions hold true, then  $f$  is **bijective**.

Let  $a \in A$  and  $b \in B$ . The **update**  $f[b/a] : A \rightarrow B$  of  $f$  at  $a$  by  $b$ , the **restriction**  $f|_C : C \rightarrow B$  of  $f$  to  $C$  and the **restricted equality**  $=_C \subseteq B^A \times B^A$  are defined as follows: For all  $a' \in A$  and  $c \in C$ ,

$$f[b/a](a') = \begin{cases} b & \text{if } a' = a, \\ f(a') & \text{otherwise,} \end{cases}$$

$$f|_C(c) = f(c),$$

$$f =_C g \Leftrightarrow f|_{A \setminus C} = g|_{A \setminus C}.$$

$\text{inc}_C : C \rightarrow A$  denotes the **inclusion function** that maps every element of  $C$  to itself.

$\text{set2fun} : \mathcal{P}(A) \rightarrow 2^A$  maps  $C \subseteq A$  to the **indicator** or **characteristic function** of  $C$  that maps all elements of  $C$  to 1 and all elements of  $A \setminus C$  to 0.

$\text{set2fun}$  is bijective.

$\text{rel2fun} : \mathcal{P}(A \times B) \rightarrow \mathcal{P}(B)^A$  transforms binary relations into multivalued functions: For all  $a \in A$  and  $R \subseteq A \times B$ ,  $\text{rel2fun}(R)$  maps  $a$  to  $\{b \in B \mid (a, b) \in R\}$ .

*rel2fun* is bijective.

Since  $A \times 1 \cong A$  and  $\mathcal{P}(1) \cong 2$ , *rel2fun* with  $B = 1$  yields *set2fun*.

Let  $A = B$ . Then  $f^0 =_{def} id_A$  and for all  $n > 0$ ,  $f^{n+1} =_{def} f^n \circ f$ .

For all  $i \in I$ , let  $A_i$  be a set.  $\bigtimes_{i \in I} A_i$  denotes the **Cartesian product** of all  $A_i$ :

$$\bigtimes_{i \in I} A_i =_{def} \{f : I \rightarrow \bigcup_{i \in I} A_i \mid \forall i \in I : f(i) \in A_i\}.$$

An element  $f \in \bigtimes_{i \in I} A_i$  is called an  **$I$ -tuple** and often written as  $(f(i))_{i \in I}$  or  $(f(i))_{i \in J}$  where  $J = \{i \in I \mid A_i \neq 1\}$ .

For all  $n > 0$ ,  $A_1 \times \cdots \times A_n =_{def} \bigtimes_{i=1}^n A_i =_{def} \bigtimes_{i \in [n]} A_i$ .

$\biguplus_{i \in I} A_i$  denotes the **disjoint union** of all  $A_i$ :

$$\biguplus_{i \in I} A_i =_{def} \{(a, i) \mid a \in A_i, i \in I\}.$$

$(a, i) \in \biguplus_{i \in I} A_i$  is also written as  $i(a)$ .

For all  $n > 0$ ,  $A_1 + \cdots + A_n =_{\text{def}} \biguplus_{i=1}^n A_i =_{\text{def}} \biguplus_{i \in [n]} A_i$ .

The universal properties of products and sums provide a good introduction into category-theoretical thinking, i.e., reasoning in terms of equations between morphisms, here: functions.

Every data model is a product, a sum, a subset, intuitively: a *restriction*, of a product or a *quotient*, intuitively: an *abstraction*, of a sum (see, e.g., chapter 6).

## Products

Let  $A = (A_i)_{i \in I}$  be a tuple of sets,  $P$  be a set and  $\pi = (\pi_i : P \rightarrow A_i)_{i \in I}$  be a tuple of functions. Since all functions have the same domain, such a tuple is called a **cone**.

The pair  $(P, \pi)$ , also written as  $\prod_{i \in I} A_i$ , is called a **product of  $A$**  if for all  $(f_i : B \rightarrow A_i)_{i \in I}$  there is a unique function  $f : B \rightarrow P$  such that for all  $i \in I$ ,

$$\pi_i \circ f = f_i. \tag{1}$$

$\pi_i$  is called the  **$i$ -th projection of  $P$**  and  $f$  the **product extension or range tupling of  $(f_i)_{i \in I}$  to  $P$** . Since  $f$  is determined by  $(f_i)_{i \in I}$ , we write  $\langle f_i \rangle_{i \in I}$  for  $f$ .

Consequently, for all  $f, g : B \rightarrow P$ ,

$$(\forall i \in I : \pi_i \circ f = \pi_i \circ g) \Rightarrow f = g. \quad (2)$$

All products of  $A$  are isomorphic to each other:

Let  $(P, \pi)$  and  $(P', \pi')$  be products of  $A$  with tupling constructors  $\langle \rangle$  and  $\langle \rangle'$ , respectively. Then  $\langle \pi'_i \rangle_{i \in I} : P' \rightarrow P$  is bijective with inverse  $\langle \pi_i \rangle'_{i \in I}$ .

*Proof.* For all  $i \in I$ , let  $f_i = \pi'_i \circ \langle \pi_i \rangle'_{i \in I} : P \rightarrow A_i$  and  $f'_i = \pi_i \circ \langle \pi'_i \rangle_{i \in I} : P' \rightarrow A_i$ . Since for all  $i \in I$ ,  $f = \langle \pi'_i \rangle_{i \in I} \circ \langle \pi_i \rangle'_{i \in I} : P \rightarrow P$  and  $f = id_P$  satisfy (1), both functions agree with each other, i.e.,

$$\langle \pi'_i \rangle_{i \in I} \circ \langle \pi_i \rangle'_{i \in I} = id_P. \quad (3)$$

Since for all  $i \in I$ ,  $f = \langle \pi_i \rangle'_{i \in I} \circ \langle \pi'_i \rangle_{i \in I} : P' \rightarrow P'$  and  $f = id_{P'}$  satisfy  $\pi'_i \circ f = f'_i$ , both functions agree with each other, i.e.,

$$\langle \pi_i \rangle'_{i \in I} \circ \langle \pi'_i \rangle_{i \in I} = id_{P'}. \quad (4)$$

By (3) and (4),  $\langle \pi_i \rangle'_{i \in I} : P \rightarrow P'$  is bijective with inverse  $\langle \pi'_i \rangle_{i \in I} : P' \rightarrow P$ . □

All sets that are isomorphic to a product of  $A$  are products of  $A$ .

*Proof.* Let  $(P, \pi)$  be a product of  $A$ ,  $P'$  be a set,  $h : P' \rightarrow P$  be a bijection and  $\pi' = (\pi_i \circ h)_{i \in I}$ . Let  $f = (f_i : B \rightarrow A_i)_{i \in I}$  and  $g = h^{-1} \circ \langle f_i \rangle_{i \in I} : B \rightarrow P'$ .

Then for all  $i \in I$ ,

$$\pi'_i \circ g = \pi_i \circ h \circ g = \pi_i \circ h \circ h^{-1} \circ \langle f_i \rangle_{i \in I} = \pi_i \circ \langle f_i \rangle_{i \in I} = f_i.$$

$g$  is unique: Let  $g' : B \rightarrow P'$  satisfy  $\pi'_i \circ g' = f_i$  for all  $i \in I$ . Then

$$\pi_i \circ h \circ g = \pi'_i \circ g = f_i = \pi'_i \circ g' = \pi_i \circ h \circ g'$$

and thus by (2),  $h \circ g = h \circ g'$ . Hence  $g = h^{-1} \circ h \circ g = h^{-1} \circ h \circ g' = g'$ .  $\square$

The Cartesian product  $\bigtimes_{i \in I} A_i$  is a product of  $A$ .

The projections and range tuplings for  $\bigtimes_{i \in I} A_i$  are defined as follows:

- For all  $i \in I$  and  $f \in \bigtimes_{i \in I} A_i$ ,  $\pi_i(f) =_{def} f(i)$ .
- For all  $(f_i : B \rightarrow A_i)_{i \in I}$ ,  $b \in B$  and  $i \in I$ ,  $\langle f_i \rangle_{i \in I}(b)(i) =_{def} f_i(b)$ .

For all  $f = (f_i : A_i \rightarrow B_i)_{i \in I}$ ,

$$\prod_{i \in I} f_i =_{def} \langle f_i \circ \pi_i \rangle_{i \in I}$$

is called the **product** of  $f$ .

For all  $f : A \rightarrow B$ , nonempty sets  $I$ ,  $n > 0$  and  $(f_i : A_i \rightarrow B_i)_{i=1}^n$ ,

$$\begin{aligned}\textcolor{red}{f^I} &=_{def} \prod_{i \in I} f, \\ \textcolor{red}{f_1 \times \cdots \times f_n} &=_{def} \prod_{i \in [n]} f_i.\end{aligned}$$

For all functions  $f : \prod_{i \in I} B_i \rightarrow B$ , sets  $A$  and  $g = (g_i : A \rightarrow B_i)_{i \in I}$ ,

$$\textcolor{red}{lift^A(f)(g)} =_{def} f \circ \langle g_i \rangle_{i \in I} : \prod_{i \in I} B_i^A \rightarrow B^A.$$

## Product equations

For all  $f : A \rightarrow B$ ,  $(f_i : B \rightarrow B_i)_{i \in I}$ ,  $(g_i : A_i \rightarrow B_i)_{i \in I}$ ,  $k \in I$  and  $(h_i : B_i \rightarrow A_i)_{i \in I}$ ,

$$\langle \pi_i \rangle_{i \in I} = id_{\prod_{i \in I} A_i}, \tag{5}$$

$$\langle f_i \rangle_{i \in I} \circ f = \langle f_i \circ f \rangle_{i \in I}, \tag{6}$$

$$\pi_k \circ \prod_{i \in I} g_i = g_k \circ \pi_k, \tag{7}$$

$$\prod_{i \in I} h_i \circ \langle f_i \rangle_{i \in I} = \langle h_i \circ f_i \rangle_{i \in I}, \tag{8}$$

$$ker(\langle f_i \rangle_{i \in I}) = \bigcap_{i \in I} ker(f_i). \tag{9}$$

## Product characterizations

Let  $d = (d_i : P \rightarrow A_i)_{i \in I}$ .

$(P, d)$  is a product of  $A$  iff for all  $a, b \in P$ ,  $a, b \in P$  are equal iff for all  $i \in I$ ,  $d_i(a) = d_i(b)$ .

*Proof.* “ $\Rightarrow$ ”: Let  $(P, d)$  be a product of  $A$ ,  $f = \lambda x.a : 1 \rightarrow P$  and  $g = \lambda x.b : 1 \rightarrow P$ . Then

$$\forall i \in I : d_i(a) = d_i(b) \Rightarrow \forall i \in I : d_i \circ f = d_i \circ g \stackrel{(2)}{\Rightarrow} f = g \Rightarrow a = f(\epsilon) = g(\epsilon) = b.$$

“ $\Leftarrow$ ”: Suppose that for all  $a, b \in P$ ,  $a, b \in P$  are equal iff for all  $i \in I$   $d_i(a) = d_i(b)$ .

Let  $(f_i : B \rightarrow A_i)_{i \in I}$ . Then  $g : B \rightarrow P$  with  $d_i(g(b)) = f_i(b)$  for all  $i \in I$  and  $b \in B$  is a product extension of  $(f_i)_{i \in I}$  because every  $f : B \rightarrow P$  that satisfies (1) agrees with  $g$ .  $\square$

$(P, d)$  is a product of  $(A_i)_{i \in I}$  iff  $\langle d_i \rangle_{i \in I} : P \rightarrow \prod_{i \in I} A_i$  is iso.

*Proof.* The “ $\Rightarrow$ ”-direction is shown above.

“ $\Leftarrow$ ”: Let  $\langle d_i \rangle_{i \in I}$  be iso and  $(f_i : B \rightarrow A_i)_{i \in I}$ . Then for all  $i \in I$ ,

$$d_i \circ \langle d_i \rangle_{i \in I}^{-1} \circ \langle f_i \rangle_{i \in I} = \pi_i \circ \langle d_i \rangle_{i \in I} \circ \langle d_i \rangle_{i \in I}^{-1} \circ \langle f_i \rangle_{i \in I} = \pi_i \circ \langle f_i \rangle_{i \in I} = f_i.$$

Hence  $f =_{def} \langle d_i \rangle_{i \in I}^{-1} \circ \langle f_i \rangle_{i \in I} : B \rightarrow P$  satisfies  $d_i \circ f = f_i$ .

Moreover,  $f$  is unique w.r.t. this property: Suppose that  $f, g : B \rightarrow P$  satisfy  $d_i \circ f = f_i = d_i \circ g$  for all  $i \in I$ . Then

$$\pi_i \circ \langle d_i \rangle_{i \in I} \circ f = d_i \circ f = d_i \circ g = \pi_i \circ \langle d_i \rangle_{i \in I} \circ g$$

and thus  $\langle d_i \rangle_{i \in I} \circ f = \langle d_i \rangle_{i \in I} \circ g$ . Hence  $f = g$  because  $\langle d_i \rangle_{i \in I}$  is iso.  $\square$

$(P, d)$  is a product of  $(A_i)_{i \in I}$  iff for all  $(f_i : B \rightarrow A_i)_{i \in I}$  there is a function  $\langle f_i \rangle_{i \in I} : B \rightarrow P$  such that for all  $i \in I$  and  $f : A \rightarrow P$  the following conditions hold true:

$$d_i \circ \langle f_i \rangle_{i \in I} = f_i, \tag{10}$$

$$\langle d_i \circ f \rangle_{i \in I} = f. \tag{11}$$

*Proof.* “ $\Leftarrow$ ”: Let  $(f_i : B \rightarrow A_i)_{i \in I}$  and suppose that some  $\langle f_i \rangle_{i \in I} : B \rightarrow P$  satisfies (10) and (11). Let  $f, g : A \rightarrow P$  satisfy  $d_i \circ f = f_i = d_i \circ g$  for all  $i \in I$ . Then

$$f \stackrel{(11)}{=} \langle d_i \circ f \rangle_{i \in I} = \langle d_i \circ g \rangle_{i \in I} \stackrel{(11)}{=} g.$$

Hence  $\langle f_i \rangle_{i \in I}$  is unique w.r.t. (10), i.e.,  $(P, d)$  is a product of  $(A_i)_{i \in I}$ .

“ $\Rightarrow$ ”: Let  $(P, d)$  be a product of  $(A_i)_{i \in I}$ . Then (10) holds true. Moreover, for all  $f : A \rightarrow P$ ,

$$\langle d_i \circ f \rangle_{i \in I} \stackrel{(6)}{=} \langle d_i \rangle_{i \in I} \circ f \stackrel{(5)}{=} id_P \circ f = f,$$

i.e., (11) holds true.  $\square$

## Sums

Let  $A = (A_i)_{i \in I}$  be a tuple of sets,  $S$  be a set and  $\iota = (\iota_i : A_i \rightarrow S)_{i \in I}$  be a tuple of functions. Since all functions have the same range, such a tuple is called a **cocone**.

The pair  $(S, \iota)$ , also written as  $\coprod_{i \in I} A_i$ , is called a **sum** or **coproduct of  $A$**  if for all  $(f_i : A_i \rightarrow B)_{i \in I}$  there is a unique function  $f : S \rightarrow B$  such that for all  $i \in I$ ,

$$f \circ \iota_i = f_i. \quad (12)$$

$\iota_i$  is called the  **$i$ -th injection of  $S$**  and  $f$  the **sum extension or source tupling of  $(f_i)_{i \in I}$  to  $S$** . Since  $f$  is determined by  $(f_i)_{i \in I}$ , we write  $[f_i]_{i \in I}$  for  $f$ .

Consequently, for all  $f, g : S \rightarrow B$ ,

$$(\forall i \in I : f \circ \iota_i = g \circ \iota_i) \Rightarrow f = g. \quad (13)$$

All sums of  $A$  are isomorphic to each other:

Let  $(S, \iota)$  and  $(S', \iota')$  be sums of  $A$  with tupling constructors  $[ ]$  and  $[ ]'$ , respectively. Then  $[\iota'_i]_{i \in I} : S \rightarrow S'$  is bijective with inverse  $[\iota_i]_{i \in I}'$ .

*Proof.* For all  $i \in I$ , let  $f_i = [\iota_i]_{i \in I}' \circ \iota'_i : A_i \rightarrow S$  and  $f'_i = [\iota'_i]_{i \in I} \circ \iota_i : A_i \rightarrow S'$ .

Since for all  $i \in I$ ,  $f = [\iota_i]_{i \in I}' \circ [\iota'_i]_{i \in I} : S \rightarrow S$  and  $f = id_S$  satisfy (12), both functions agree with each other, i.e.,

$$[\iota_i]_{i \in I}' \circ [\iota'_i]_{i \in I} = id_S. \quad (14)$$

Since for all  $i \in I$ ,  $f = [\iota'_i]_{i \in I} \circ [\iota_i]_{i \in I}' : S' \rightarrow S'$  and  $f = id_{S'}$  satisfy  $f \circ \iota'_i = f'_i$ , both functions agree with each other, i.e.,

$$[\iota'_i]_{i \in I} \circ [\iota_i]_{i \in I}' = id_{S'}. \quad (15)$$

By (14) and (15),  $[\iota'_i]_{i \in I} : S \rightarrow S'$  is bijective with inverse  $[\iota_i]_{i \in I}' : S' \rightarrow S$ . □

All sets that are isomorphic to a sum of  $A$  are sums of  $A$ .

*Proof.* Let  $(S, \iota)$  be a sum of  $A$ ,  $S'$  be a set,  $h : S \rightarrow S'$  be a bijection and  $\iota' = (h \circ \iota_i)_{i \in I}$ . Let  $f = (f_i : A_i \rightarrow B)_{i \in I}$  and  $g = [f_i]_{i \in I} \circ h^{-1} : S' \rightarrow B$ . Then for all  $i \in I$ ,

$$g \circ \iota'_i = g \circ h \circ \iota_i = [f_i]_{i \in I} \circ h^{-1} \circ h \circ \iota_i = [f_i]_{i \in I} \circ \iota_i = f_i.$$

$g$  is unique: Let  $g' : S' \rightarrow B$  satisfy  $g' \circ \iota_i = f_i$  for all  $i \in I$ . Then

$$g \circ h \circ \iota_i = g \circ \iota'_i = f_i = g' \circ \iota'_i = g' \circ h \circ \iota_i$$

and thus by (12),  $g \circ h = g' \circ h$ . Hence  $g = g \circ h \circ h^{-1} = g' \circ h \circ h^{-1} = g'$ .  $\square$

The disjoint union  $\biguplus_{i \in I} A_i$  is a sum of  $A$ .

The injections and source tuplings for  $\biguplus_{i \in I} A_i$  are defined as follows:

- For all  $i \in I$  and  $a \in A_i$ ,  $\iota_i(a) =_{def} (a, i)$ .
- For all  $(f_i : A_i \rightarrow B)_{i \in I}$ ,  $i \in I$  and  $a \in A_i$ ,  $[f_i]_{i \in I}(a, i) =_{def} f_i(a)$ .

For all  $(f_i : A_i \rightarrow B_i)_{i \in I}$ ,

$$\coprod_{i \in I} f_i =_{def} [\iota_i \circ f_i]_{i \in I}$$

is called the **sum** or **coproduct** of  $f$ .

For all nonempty sets  $I$ ,  $f : A \rightarrow B$ ,  $n > 0$  and  $(f_i : A_i \rightarrow B_i)_{i=1}^n$ ,

$$\begin{aligned} f \times I &=_{def} \coprod_{i \in I} f, \\ f_1 + \cdots + f_n &=_{def} \coprod_{i \in [n]} f_i. \end{aligned}$$

For all functions  $f : B \rightarrow \coprod_{i \in I} B_i$ , sets  $A$  and  $g = (g_i : B_i \rightarrow A)_{i \in I}$ ,

$$\text{lift}_A(f)(g) =_{\text{def}} [g_i]_{i \in I} \circ f : \prod_{i \in I} A^{B_i} \rightarrow A^B.$$

## Sum equations

For all  $(f_i : A_i \rightarrow A)_{i \in I}$ ,  $f : A \rightarrow B$ ,  $(g_i : A_i \rightarrow B_i)_{i \in I}$ ,  $k \in I$  and  $(h_i : B_i \rightarrow A_i)_{i \in I}$ ,

$$[\iota_i]_{i \in I} = id_{\coprod_{i \in I} A_i}, \quad (16)$$

$$f \circ [f_i]_{i \in I} = [f \circ f_i]_{i \in I}, \quad (17)$$

$$\coprod_{i \in I} g_i \circ \iota_k = \iota_k \circ g_k, \quad (18)$$

$$[f_i]_{i \in I} \circ \coprod_{i \in I} h_i = [f_i \circ h_i]_{i \in I}, \quad (19)$$

$$img([f_i]_{i \in I}) = \bigcup_{i \in I} img(f_i). \quad (20)$$

## Sum characterizations

Let  $(c_i : A_i \rightarrow S)_{i \in I}$ .

$(S, c)$  is a sum of  $(A_i)_{i \in I}$  iff for all  $a \in S$  there are unique  $i \in I$  and  $b \in A_i$  with  $c_i(b) = a$ .

*Proof.* “ $\Rightarrow$ ”: Let  $(S, c)$  be a sum of  $(A_i)_{i \in I}$ . Assume that there is  $a \in S \setminus \bigcup_{i \in I} c_i(A_i)$ . Let  $f, g : S \rightarrow 2$  be defined as follows:  $f = \lambda x.0$  and  $g = (\lambda x.\text{if } x \in S \setminus \{a\} \text{ then } 0 \text{ else } 1)$ . Then for all  $i \in I$  and  $b \in A_i$ ,

$$(f \circ c_i)(b) = f(c_i(b)) = 0 = g(c_i(b)) = (g \circ c_i)(b),$$

and thus by (13),  $f = g$ .  $\ntriangleleft$  Hence  $S = \bigcup_{i \in I} c_i(A_i)$ .

For all  $i \in I$ , define  $f_i : A_i \rightarrow \biguplus_{i \in I} A_i$  as follows: For all  $a \in A_i$ ,  $f_i(a) = (a, i)$ . Then for all  $i, j \in I$ ,  $a \in A_i$  and  $b \in A_j$ ,  $c_i(a) = c_j(b)$  implies

$$(a, i) = f_i(a) = [f_i]_{i \in I}(\iota_i(a)) = [f_i]_{i \in I}(\iota_j(b)) = f_j(b) = (b, j).$$

“ $\Leftarrow$ ”: Suppose that for all  $a \in S$  there are unique  $i \in I$  and  $b \in A_i$  with  $\iota_i(b) = a$ .

Let  $(f_i : A_i \rightarrow B)_{i \in I}$ . Then  $g : S \rightarrow B$  with  $g(c_i(b)) = f_i(b)$  for all  $i \in I$  and  $b \in A_i$  is a sum extension of  $(f_i)_{i \in I}$  because every  $f : S \rightarrow B$  that satisfies (12) agrees with  $g$ .  $\square$

$(S, c)$  is a sum of  $(A_i)_{i \in I}$  iff  $[c_i]_{i \in I} : \coprod_{i \in I} A_i \rightarrow S$  is iso.

*Proof.* The “ $\Rightarrow$ ”-direction is shown above.

“ $\Leftarrow$ ”: Let  $[c_i]_{i \in I}$  be iso and  $(f_i : A_i \rightarrow B)_{i \in I}$ . Then for all  $i \in I$ ,

$$[f_i]_{i \in I} \circ [c_i]_{i \in I}^{-1} \circ c_i = [f_i]_{i \in I} \circ [c_i]_{i \in I}^{-1} \circ [c_i]_{i \in I} \circ \iota_i = [f_i]_{i \in I} \circ \iota_i = f_i.$$

Hence  $f =_{def} [f_i]_{i \in I} \circ [c_i]_{i \in I}^{-1} : S \rightarrow B$  satisfies  $f \circ c_i = f_i = g \circ c_i$ . Then

$$f \circ [c_i]_{i \in I} \circ \iota_i = f \circ c_i = g \circ c_i = g \circ [c_i]_{i \in I} \circ \iota_i$$

and thus  $f \circ [c_i]_{i \in I} = g \circ [c_i]_{i \in I}$ . Hence  $f = g$  because  $[c_i]_{i \in I}$  is iso.  $\square$

$(S, c)$  is a sum of  $(A_i)_{i \in I}$  iff for all  $(f_i : A_i \rightarrow B)_{i \in I}$  there is  $[f_i]_{i \in I} : S \rightarrow B$  such that for all  $i \in I$  and  $f : S \rightarrow A$  the following conditions hold true:

$$[f_i]_{i \in I} \circ c_i = f_i, \tag{21}$$

$$[f \circ c_i]_{i \in I} = f. \tag{22}$$

*Proof.* “ $\Leftarrow$ ”: Let  $(f_i : A_i \rightarrow B)_{i \in I}$  and suppose that some  $[f_i]_{i \in I} : S \rightarrow B$  satisfies (21) and (22). Let  $f, g : S \rightarrow A$  satisfy  $f \circ c_i = f_i = g \circ c_i$  for all  $i \in I$ . Then

$$f \stackrel{(22)}{=} [f \circ c_i]_{i \in I} = [g \circ c_i]_{i \in I} \stackrel{(22)}{=} g.$$

Hence  $[f_i]_{i \in I}$  is unique w.r.t. (21), i.e.,  $(S, c)$  is a sum of  $(A_i)_{i \in I}$ .

“ $\Leftarrow$ ”: Let  $(S, c)$  be a sum of  $(A_i)_{i \in I}$ . Then (21) holds true. Moreover, for all  $f : S \rightarrow A$ ,

$$[f \circ c_i]_{i \in I} \stackrel{(17)}{=} f \circ [c_i]_{i \in I} \stackrel{(16)}{=} f \circ id_S = f,$$

i.e., (22) holds true. □

## Miscellanea

The sets of **nonempty** or all **words** or **finite lists over**  $A$  are defined as follows:

$$A^+ =_{\text{def}} \bigcup_{n>0} A^n,$$

$$A^* =_{\text{def}} A^+ \cup \{\epsilon\}.$$

$(a_1, \dots, a_n) \in A^n$  is often written as  $a_1 \dots a_n$ .

$\epsilon$  denotes the empty word and is supposed to be different from any element of a set of symbols like the alphabet  $\mathcal{I}$  of a set of types over  $(S, \mathcal{I})$  (see chapter 7).

$|\epsilon| =_{\text{def}} 0$ . For all  $n > 0$  and  $w \in A^n$ ,  $|w| =_{\text{def}} n$ .

$A^{\mathbb{N}}$  is called the set of **streams** or **infinite lists over**  $A$ .

$A^{\infty} =_{\text{def}} A^* \cup A^{\mathbb{N}}$  is called the set of **colists over**  $A$ .

For all sets  $A, B$ ,  $\Omega : A \rightarrow B + 1$  denotes the **nowhere-defined function**, i.e., for all  $a \in A$ ,  $\Omega(a) = ()$ .

## Functions and relations on words and streams

Let  $A$  be a set. For all  $v = (a_1, \dots, a_m)$ ,  $w = (b_1, \dots, b_n) \in A^+$  and  $f \in A^{\mathbb{N}}$ ,

$$\text{head}(\epsilon) =_{\text{def}} (),$$

$$\text{head}(v) =_{\text{def}} a_1,$$

$$\text{head}(f) =_{\text{def}} f(0),$$

$$\text{tail}(\epsilon) =_{\text{def}} (),$$

$$\text{tail}(v) =_{\text{def}} \text{if } m = 1 \text{ then } \epsilon \text{ else } (a_2, \dots, a_m),$$

$$\text{tail}(f) =_{\text{def}} \lambda n. f(n + 1),$$

$$\epsilon \cdot \epsilon =_{\text{def}} \epsilon,$$

$$w \cdot \epsilon =_{\text{def}} w,$$

$$\epsilon \cdot w =_{\text{def}} w,$$

$$\epsilon \cdot f =_{\text{def}} f,$$

$$v \cdot w =_{\text{def}} (a_1, \dots, a_m, b_1, \dots, b_n),$$

$$v \cdot f =_{\text{def}} \lambda i. \text{if } i < m \text{ then } a_{i+1} \text{ else } f(m - i),$$

$$\epsilon^{-1} =_{\text{def}} \epsilon,$$

$$v^{-1} =_{\text{def}} (a_m, \dots, a_1).$$

The concatenation operator  $\cdot$  binds stronger than other binary word operator and is often omitted.

For all  $B \subseteq A^*$  and  $C \subseteq A^\infty$ ,

$$\begin{aligned} B \cdot C &=_{def} \{a \cdot b \mid a \in B, b \in C\}, \\ B^{-1} &=_{def} \{a^{-1} \mid a \in B\}. \end{aligned}$$

$v \in A^*$  is a **prefix** of  $w \in A^*$  if  $w = v \cdot v'$  for some  $v' \in A^*$ .

The binary **prefix relation**  $\leq$  on  $A^*$  is the set of all pairs  $(v, w) \in (A^*)^2$  such that  $v$  is a prefix of  $w$ .

For all  $v, w \in A^*$  and  $a \in A$ ,  $\#a(w)$  denotes the number of occurrences of  $a$  in  $w$ ,

$v =_{bag} w \Leftrightarrow_{def} v$  is a **permutation** of  $w$ , i.e., for all  $a \in A$ ,  $\#a(v) = \#a(w)$ ,

$v =_{set} w \Leftrightarrow_{def}$  for all  $a \in A$ ,  $\#a(v) > 0 \Leftrightarrow \#a(w) > 0$ .

Let  $A, B, C$  be sets,  $g : A \rightarrow B$ ,  $h : A \rightarrow A + B$ ,  $n \in \mathbb{N}$  and  $p : A \rightarrow 2$ .

$$\begin{aligned} g^+ &: A^+ \rightarrow B^+ \\ (a_1, \dots, a_n) &\mapsto (g(a_1), \dots, g(a_n)) \end{aligned}$$

$$\begin{aligned} g^* &: A^* \rightarrow B^* \\ w &\mapsto \text{if } w = \epsilon \text{ then } \epsilon \text{ else } g^+(s) \end{aligned}$$

$$\begin{aligned} h^n &: A \rightarrow A + B \\ a &\mapsto \begin{cases} a & \text{if } n = 0 \\ h(a') & \text{if } n > 0 \wedge a' = h^{n-1}(a) \in A \\ b & \text{if } n > 0 \wedge b = h^{n-1}(a) \in B \end{cases} \end{aligned}$$

$$\begin{aligned} h^C &: A^C \rightarrow B^C \\ f &\mapsto h \circ f \end{aligned}$$

$$\begin{aligned} p? &: A \rightarrow A + A \\ a &\mapsto \text{if } p(a) = 1 \text{ then } \iota_1(a) \text{ else } \iota_2(a) \end{aligned}$$

$$\begin{aligned} pair &: A \rightarrow (A \times B)^B \\ a &\mapsto \lambda b.(a, b) \end{aligned}$$

$$\begin{aligned} \text{curry} : C^{A \times B} &\rightarrow (C^B)^A \\ f &\mapsto \lambda a. \lambda b. f(a, b) = f^B \circ \text{pair} \end{aligned}$$

$$\begin{aligned} \text{apply} : B^A \times A &\rightarrow B \\ (f, a) &\mapsto f(a) \end{aligned}$$

$$\begin{aligned} \text{uncurry} : (C^B)^A &\rightarrow C^{A \times B} \\ f &\mapsto \lambda(a, b). f(a, b) = \text{apply} \circ (f \times \text{id}_B) \end{aligned}$$

$$\begin{aligned} \text{flip} : (C^B)^A &\rightarrow (C^A)^B \\ f &\mapsto = \lambda b. \lambda a. f(a)(b) \end{aligned}$$

$f : A^{\mathbb{N}} \rightarrow B^{\mathbb{N}}$  is **causal** if for all  $s, s' \in A^{\mathbb{N}}$  and  $n \in \mathbb{N}$ ,

$$(s(0), \dots, s(n)) = (s'(0), \dots, s'(n)) \text{ implies } f(s)(n) = f(s')(n).$$

The set  $\mathcal{C}(A, B)$  of causal functions from  $A^{\mathbb{N}}$  to  $B^{\mathbb{N}}$  is isomorphic to the set of functions from  $A^+$  to  $B$ .

*Proof.* Let  $g : B^{A^+} \rightarrow \mathcal{C}(A, B)$  and  $h : \mathcal{C}(A, B) \rightarrow B^{A^+}$  be defined as follows:

- For all  $f : A^+ \rightarrow B$ ,  $s \in A^{\mathbb{N}}$  and  $n \in \mathbb{N}$ ,  $g(f)(s)(n) = f(s(0), \dots, s(n))$ .

- For all  $f' \in \mathcal{C}(A, B)$ ,  $n \in \mathbb{N}$ ,  $(a_1, \dots, a_{n+1}) \in A^+$  and  $s \in A^{\mathbb{N}}$ ,

$$(s(0), \dots, s(n)) = (a_1, \dots, a_{n+1}) \quad \text{implies} \quad h(f')(a_1, \dots, a_{n+1}) = f'(s)(n).$$

Since  $f$  is causal,  $h$  is well-defined. Moreover, for all  $f : A^+ \rightarrow B$ ,  $s \in A^{\mathbb{N}}$ ,  $f' \in \mathcal{C}(A, B)$  and  $n \in \mathbb{N}$ ,

$$\begin{aligned} h(g(f))(s(0), \dots, s(n)) &= g(f)(s)(n) = f(s(0), \dots, s(n)), \\ g(h(f'))(s)(n) &= h(f')(s(0), \dots, s(n)) = f'(s)(n). \end{aligned}$$

Hence  $g$  is bijective. □

Let  $h : A \rightarrow B$ .

$$\mathcal{P}_\omega(A) =_{def} \{C \subseteq A \mid C \text{ is finite}\}.$$

$\mathcal{P}(h) : \mathcal{P}(A) \rightarrow \mathcal{P}(B)$  and  $\mathcal{P}_\omega(h) : \mathcal{P}_\omega(A) \rightarrow \mathcal{P}_\omega(B)$  map every (finite) subset  $C$  of  $A$  to  $\{h(c) \mid c \in C\}$ .

$\mathcal{P}(h)(C)$  and  $\mathcal{P}_\omega(h)(C)$  are often abbreviated to  $h(C)$ .

$supp(h) = \{a \in A \mid h(a) \notin \{0, \emptyset, \epsilon\}\}$  is called the **support** of  $h$ .

If  $supp(h)$  is finite, then  $h$  is called a  **$B$ -weighted set** with weights from  $B$ .

$A \rightarrow_{\omega} B$  and  $B_{\omega}^A$  denote the set of  $B$ -weighted sets with domain  $A$ .

$\mathbb{N}^A$  is called the set of **bags** or **multisets** of elements of  $A$ .

$\mathbb{N}_{\omega}^A$  is called the set of **finite bags** or **finite multisets** of elements of  $A$ .

$$2_{\omega}^A \cong \mathcal{P}_{\omega}(A).$$

Let  $(M, +, 0)$  be a commutative monoid and  $C \subseteq M$ .

$M_{\omega}^h : M_{\omega}^A \rightarrow M_{\omega}^B$  is defined as follows: For all  $f \in M_{\omega}^A$  and  $b \in B$ ,

$$M_{\omega}^h(f)(b) = \sum \{f(a) \mid a \in A, h(a) = b\}. \quad (1)$$

$M_{\omega}^h$  is well-defined: For all  $f \in M_{\omega}^A$  and  $b \in B$ ,

$$\begin{aligned} b \in supp(M_{\omega}^h(f)) &\stackrel{(1)}{\Rightarrow} \sum \{f(a) \mid h(a) = b\} \neq 0 \Rightarrow \exists a \in A : f(a) \neq 0 \wedge h(a) = b \\ &\Rightarrow \exists a \in supp(f) : h(a) = b \Rightarrow b \in h(supp(f)). \end{aligned}$$

Hence  $|supp(M_{\omega}^h(f))| \leq |h(supp(f))| \leq |supp(f)| < \omega$ , i.e.,  $M_{\omega}^h(f)$  has finite support and thus belongs to  $M_{\omega}^B$ .

$M_C^A =_{def} \{f \in M_\omega^A \mid \sum_{a \in A} f(a) \in C\}$  is called the set of **constrained  $M$ -weighted sets** with domain  $A$  and constraint  $\varphi$  (see [156], section 7).

$M_C^h : M_C^A \rightarrow M_C^B$  denotes the restriction of  $M_\omega^h$  to  $M_C^A$ .

$M_C^h$  is well-defined:  $|supp(M_C^h(f))| < \omega$  can be proved along the lines of the above proof that  $M_\omega^h(f)$  has finite support. Moreover,

$$\sum_{b \in B} M_C^h(f)(b) \stackrel{(1)}{=} \sum_{b \in B} \sum_{a \in A, h(a)=b} f(a) = \sum_{a \in A, h(a) \in B} f(a) = \sum_{a \in A} f(a) \in C.$$

Hence  $M_C^h(f) \in M_C^B$ .

Given a  $f \in M_\omega^A$  with  $supp(f) = \{a_1, \dots, a_n\}$  and values  $m_i = f(a_i)$  for all  $1 \leq i \leq m$ ,  $f$  is often denoted by the expression

$$\sum_{i=1}^n m_i \cdot a_i$$

where  $a_1, \dots, a_n$  are regarded as variables constants.

This notation also allows us to define  $M_C^h$  (and  $M_\omega^h$ ) simply as follows:

For all  $m_1, \dots, m_n \in M$ ,

$$M_C^h\left(\sum_{i=1}^n m_i \cdot a_i\right) = \sum_{i=1}^n m_i \cdot h(a_i) \quad (2)$$

(see [77], Def. 4.1.1).

$\mathcal{D}_\omega(A) =_{def} (\mathbb{R}_{\geq 0})_{\{1\}}^A$  is called the set of (discrete) **probability distributions** of elements of  $A$ .

Since  $(\mathbb{R}_{\geq 0}, +, 0)$  is **zero-sum free**, i.e., if for all  $r, s \in \mathbb{R}_{\geq 0}$ ,  $r + s = 0$  implies  $r = 0$  and  $s = 0$ ,  $\mathcal{D}_\omega(A)$  is a subset of  $[0, 1]_\omega^A$  where  $[0, 1]$  denotes the closed interval of real numbers between 0 and 1.

$\mathcal{D}_\omega(h) : \mathcal{D}_\omega(A) \rightarrow \mathcal{D}_\omega(B)$  is defined as  $(\mathbb{R}_{\geq 0})_{\{1\}}^h$ .

$L \subseteq A^*$  is **prefix closed** if for all  $w \in A^*$  and  $a \in A$   $wa \in L$  implies  $w \in L$ .

$ltr(A, B)$  denotes the set of **labelled trees over**  $(A, B)$ , i.e., functions from  $A^*$  to  $B + 1$  with prefix closed domain.

In fact,  $t \in ltr(A, B)$  represents a tree with edge labels from  $A$  and node labels from  $B$  such that two different edges each with the same source have different labels.  $t(\epsilon)$  is the label of the root and each word  $w \in A^*$  corresponds to a path that emanates from the root and ends in a node labelled with  $t(w)$ .

If  $t(w) = () \Leftrightarrow w = \epsilon$ , then  $t$  is identified with  $t(\epsilon)$ . Otherwise  $t$  may be written as follows:

$$t(\epsilon)\{a \rightarrow \lambda w.t(aw) \mid a \in A, t(a) \neq ()\}.$$

This notation is inspired by the syntax of Haskell records, i.e., data types with attributes (field names), which come here as edge labels.

For mutually disjoint edge label predicates  $p_1, \dots, p_n : A \rightarrow 2$ ,

$$x\{a \triangleright p_1(a) \rightarrow t_1, \dots, a \triangleright p_n(a) \rightarrow t_n \mid a \in A\} \text{ stands for } x(\bigcup_{i=1}^n \{a \rightarrow t_i \mid a \in A, p_i(a)\}).$$

## Further notational conventions

For all  $b \in B + 1$  and sets  $I$ ,  $b\{i \rightarrow t_i \mid i \in I\}$  is also written as  $b(t_i)_{i \in I}$ .

For all sets  $I$ ,  $(\textcolor{blue}{\lambda})(t_i)_{i \in I}$  is also written as  $(t_i)_{i \in I}$ .

For all  $b \in B + 1$ ,  $b\{\textcolor{blue}{()}\rightarrow t\}$  is also written as  $b(t)$ .

For all  $b \in B + 1$ ,  $\textcolor{blue}{b}()$  is also written as  $b$ .

A leaf, i.e., a tree of the form  $\Omega[b/\epsilon]$ , is identified with its label  $b$ .

$t$  is **finite** if  $\text{def}(t)$  is finite.  $t$  is **infinite** if  $\text{def}(t)$  is infinite.

$t$  is **finitely branching** if for all  $w \in A^*$ ,  $\text{def}(t) \cap w \cdot A$  is finite.

$t$  is **well-founded** if there is  $n \in \mathbb{N}$  such that for all  $w \in \text{def}(t)$ ,  $|w| \leq n$ , intuitively:  $t$  has finite depth, i.e., all paths emanating from the root are finite.

$t' \in \text{ltr}(A, B)$  is a **subtree** of  $t \in \text{ltr}(A, B)$  if  $t' = \lambda w.t(vw)$  for some  $v \in X^*$ .

$t \in \text{ltr}(A, B)$  is **rational** if  $t$  has only finitely many subtrees.

If  $t \in \text{ltr}(A, B)$  is rational and finitely branching, then  $\text{def}(t)$  is a regular subset of  $A^*$ .

$t$  is rational iff  $t$  can be represented as a finite graph and thus as an *iterative equation* (see chapter 14). For instance,  $t = b\{a \rightarrow t\}$  represents the tree  $t = \lambda w.b$  with  $\text{def}(t) = \{a\}^*$ .

$ftr(A, B)$ ,  $fbtr(A, B)$ ,  $itr(A, B)$ ,  $wtr(A, B)$  and  $rtr(A, B)$  denote the sets of finite, finitely branching, infinite, well-founded and rational labelled trees over  $(A, B)$ , respectively.

Labelled trees are used in chapter 9 as representations of the elements of *initial* as well as *final* models. This works mainly because

- functions *on* sets of *well-founded* labelled trees can usually be defined by structural induction and
- the values of functions *into* sets of (even non-well-founded) labelled trees over  $(A, B)$  can usually be defined by induction on  $A^*$ .

Inductive definitions of the second kind become *coinductive* if one reformulates them in terms of non-functional representations of the labelled trees (see chapter 13).

## Relations, ordered sets, and fixpoints

Let  $A$  be a set and  $R$  be a binary relation on  $A$ , i.e.,  $R$  is a subset of  $A^2$ .

$R$  is **reflexive** if  $R$  contains the 2-dimensional diagonal of  $A$ .

$R$  is **transitive** if for all  $(a, b), (b, c) \in R$ ,  $(a, c) \in R$ .

$R$  is **symmetric** if for all  $(a, b) \in R$ ,  $(b, a) \in R$ .

$R$  is an **equivalence relation on  $A$**  if  $R$  is reflexive, transitive and symmetric. Then  $A/R =_{\text{def}} \{[a]_R \mid a \in R\}$  is called the **quotient (set) of  $A$  by  $R$**  where for all  $a \in A$ ,  $[a]_R =_{\text{def}} \{b \in A_s \mid (a, b) \in R_s\}$  is called the **equivalence class of  $A$  by  $R$** .  $\text{nat}_R : A \rightarrow A/R$  denotes the **natural function** that maps every element of  $A$  to the equivalence class it belongs to.

The **equivalence closure of  $R$** ,  $R^{eq}$ , is the least equivalence relation that contains  $R$ .

$R$  is **antisymmetric** if for all  $a, b \in A$ ,  $aRb$  and  $bRa$  implies  $a = b$ .

$R$  is a **partial order** and  $(A, R)$  is a **partially ordered set** or **poset** if  $R$  is reflexive, transitive and antisymmetric.

Let  $(A, R)$  be a poset.

$R$  is a **total order** and  $(A, R)$  is a **totally ordered set** if for all  $a, b \in A$ ,  $aRb$  or  $bRa$ . If, in addition, for all  $a \in A$ ,  $(a, a) \notin R$ , then  $R$  is **strictly total**.

$R$  is **well-founded** if every nonempty subset of  $A$  contains a minimal element w.r.t.  $R$ . If, in addition,  $R$  is total, then  $R$  is a **well-order** and, consequently, each nonempty subset of  $A$  has a *least* element w.r.t.  $R$ .

A labelled tree  $t$  over  $(A, B)$  is  **$R$ -based** if  $\epsilon \in \text{def}(t)$  and for all  $w \in A^*$  and  $a, b \in A$ ,

$$wb \in \text{def}(t) \wedge (a, b) \in R \Rightarrow wa \in \text{def}(t).$$

*otr*( $A, R, B$ ) denotes the set of all  $R$ -based labelled trees over  $(A, B)$ .

$C \subseteq A$  is a **chain** if the restriction of  $R$  to  $C$  is a total order.

$C \subseteq A$  is **directed** if every finite subset of  $C$  has a supremum in  $A$ .

Let  $(A, \leq)$  be a poset and  $\lambda$  be an ordinal number, i.e., either  $0$  or

- a successor ordinal  $n + 1 = n \cup \{n\}$  for some ordinal  $n$  or
- a limit ordinal, i.e., the set of all smaller ordinals.

For instance,  $\omega = \mathbb{N}$  is the first limit ordinal.

A subset  $\{a_i \mid i < \lambda\}$  of  $A$  is a  **$\lambda$ -chain** or  **$\lambda$ -cochain** if for all ordinals  $i < j < \lambda$ ,  $a_i \leq a_j$  or  $a_j \leq a_i$ , respectively.

A poset  $(A, \leq)$  is  **$\lambda$ -complete** or a  **$\lambda$ -CPO** if  $A$  contains a least element  $\perp_A$  w.r.t.  $\leq$  and for each  $\lambda$ -chain  $B = \{a_i \mid i < \lambda\}$  of  $A$ ,  $A$  contains the supremum  $\bigsqcup B$  of  $B$ , also written as:  $\bigsqcup_{i < \lambda} a_i$ .

A poset  $(A, \leq)$  is  **$\lambda$ -cocomplete** or a  **$\lambda$ -co-CPO** if  $A$  has a greatest element  $\top$  w.r.t.  $\leq$  and for each  $\lambda$ -cochain  $B$  of  $A$ ,  $A$  contains the infimum  $\bigsqcap B$  of  $B$ , also written as:  $\bigsqcap_{i < \lambda} a_i$ .

A poset  $(A, \leq)$  is  **$\lambda$ -bicomplete** or a  **$\lambda$ -bi-CPO** if  $A$  is both  $\lambda$ -complete and  $\lambda$ -cocomplete. Note that  $\geq =_{def} \leq^{-1}$  is a partial order iff  $\leq$  is a partial order. However, completeness w.r.t.  $\geq$  need not be the same as cocompleteness w.r.t.  $\leq$ !

## Examples

A  $\lambda$ -CPO  $(A, \leq)$  is **flat** if for all  $a, b \in A$ ,  $a \leq b$  iff  $a \in \{\perp_A, b\}$ .

$\{\perp_A\}$  and  $\{\perp_A, a\}$ ,  $a \in A$ , are the only  $\lambda$ -chains of  $A$ .

$\perp_A$  and  $a$  are the suprema of  $\{\perp_A\}$  and  $\{\perp_A, a\}$ , respectively.

For every set  $A$ , the powerset of  $A$  is a  $\lambda$ -CPO and a  $\lambda$ -co-CPO: The partial order is subset inclusion, the least element is the empty set, the greatest element is  $A$ , the supremum of a  $\lambda$ -chain is the union of the elements of the chain, and the infimum of a  $\lambda$ -cochain is the intersection of the elements of the chain.

For all sets  $A, B$ , the set  $A \rightarrow B + 1$  is a  $\lambda$ -CPO: The partial order is defined as follows:  
For all  $f, g : A \rightarrow B + 1$ ,

$$f \leq g \iff \forall a \in A : f(a) = g(a) \vee f(a) = () .$$

The least element is the nowhere-defined function  $\Omega : A \rightarrow B + 1$  and every  $\lambda$ -chain  $F \subseteq (A \rightarrow B + 1)$  has a supremum: For all  $a \in A$ ,

$$(\bigsqcup F)(a) = \begin{cases} f(a) & \text{if } \exists f \in F : f(a) \neq (), \\ () & \text{otherwise.} \end{cases}$$

A product of  $n$   $\lambda$ -CPOs is a  $\lambda$ -CPO. Partial order, least element and suprema are defined componentwise.

The set  $A \rightarrow B$  of **functions** from a set  $A$  to a  $\lambda$ -CPO  $(B, \leq_B)$  is a  $\lambda$ -CPO. The partial order is defined argumentwise: For all  $f, g : A \rightarrow B$ ,

$$f \leq g \Leftrightarrow_{\text{def}} \forall a \in A : f(a) \leq_B g(a). \quad (1)$$

The least element of  $A \rightarrow B$  is  $\lambda x. \perp_B$ . Suprema are defined argumentwise: For all  $\lambda$ -chains  $F \subseteq A \rightarrow B$  and  $a \in A$ ,

$$(\bigsqcup F)(a) =_{\text{def}} \bigsqcup_{f \in F} f(a). \quad \square \quad (2)$$

### Proposition DIR ([98], Cor. 1)

Let  $(A, \leq)$  be  $\lambda$ -CPO. For all directed subsets  $B$  of  $A$  with  $|B| \leq \lambda$ ,  $A$  contains the supremum  $\bigsqcup B$  of  $B$ .

*Proof.* We show the conjecture only for  $\lambda = \omega$  and refer to the proof of [98], Thm. 1, for the generalization to arbitrary ordinal numbers.

Let  $B$  be a countable directed subset of  $A$ . If  $B$  is a chain, then  $\bigsqcup B$  exists because  $A$  is  $\omega$ -complete. Otherwise  $B$  is infinite: If  $B$  were finite,  $B$  would contain two different maximal elements w.r.t.  $R$ , which contradicts the directedness of  $B$ .

Since  $B$  is infinite, there is a bijection  $f : \mathbb{N} \rightarrow B$ . We define subsets  $B_i$ ,  $i \in \mathbb{N}$ , of  $B$  inductively as follows:  $B_0 = \{f(0)\}$  and  $B_{i+1} = B_i \cup \{f(i), b_i\}$  where  $i = \min(f^{-1}(B \setminus B_i))$  and  $b_i$  is an upper bound of  $f(i)$  and (all elements of)  $B_i$ .  $b_i$  exists because  $B$  is directed and  $B_i \cup \{f(i)\}$  is a finite subset of  $B$ .

For all  $i \in \mathbb{N}$ ,  $B_i$  is finite and directed and thus a (countable) chain. Since  $A$  is  $\omega$ -complete,  $B_i$  contains the supremum  $\bigsqcup B_i$  of  $B_i$ . Since  $B_i \subseteq B_{i+1}$ ,  $\{\bigsqcup B_i \mid i \in \mathbb{N}\}$  is also a countable chain and thus has a supremum  $c$  in  $A$ .  $c$  is the supremum of  $C = \bigcup_{i < \omega} B_i$ : For all  $i \in \mathbb{N}$  and  $b \in B_i$ ,  $b \leq \bigsqcup B_i \leq c$ . Hence  $c$  is an upper bound of  $C$ . Let  $d$  be an upper bound of  $C$ . Then for all  $i \in \mathbb{N}$ ,  $\bigsqcup B_i \leq d$  and thus  $c \leq d$ .

Of course,  $\bigcup_{i < \omega} B_i \subseteq B$ . Conversely, let  $b \in B$ . Since for all  $i \in \mathbb{N}$ ,  $|B_i| > i$ , there is  $k \in \mathbb{N}$  with  $b \in B_k$ . Hence  $B = C$  and thus  $c = \bigsqcup B$ . □

Let  $(A, \leq)$  and  $(B, \leq')$  be posets and  $f : A \rightarrow B$ .

$f$  is **monotone** if for all  $a, b \in A$ ,

$$a \leq b \text{ implies } f(a) \leq' f(b).$$

Let  $A$  and  $B$  have least element  $\perp_A$  and  $\perp_B$ , respectively.

$f$  is **strict** if  $f(\perp_A) = \perp_B$ .

Let  $A, B$  be  $\lambda$ -CPOs.  $f : A \rightarrow B$  is  **$\lambda$ -continuous** if for all  $\lambda$ -chains  $C$  of  $A$ ,

$$f(\bigsqcup C) = \bigsqcup f(C).$$

Let  $A, B$  be  $\lambda$ -co-CPOs.  $f : A \rightarrow B$  is  **$\lambda$ -cocontinuous** if for all  $\lambda$ -cochains  $C$  of  $A$ ,

$$f(\prod C) = \prod f(C).$$

Let  $A, B$  be  $\lambda$ -bi-CPOs.  $f : A \rightarrow B$  is  **$\lambda$ -bicontinuous** if  $f$  is both  $\lambda$ -continuous and  $\lambda$ -cocontinuous.

**Proposition** If  $f$  is  $\lambda$ -continuous or  $\lambda$ -cocontinuous, then  $f$  is monotone. □

**Proposition MON** Let  $f$  be monotone.

- (1) For all  $\lambda$ -chains  $C$  of  $A$ ,  $\bigsqcup f(C) \leq f(\bigsqcup C)$ .
- (2)  $f$  is  $\lambda$ -continuous iff for all  $\lambda$ -chains  $C$  of  $A$ ,  $f(\bigsqcup C) \leq \bigsqcup f(C)$ .
- (3)  $f$  is  $\lambda$ -cocontinuous iff for all  $\lambda$ -cochains  $C$  of  $A$ ,  $\prod f(C) \leq f(\prod C)$ .
- (4)  $f$  is  $\lambda$ -continuous if  $A$  is **chain-finite**, i.e., all  $\lambda$ -chains of  $A$  are finite.

(5)  $f$  is  $\lambda$ -cocontinuous if  $A$  is **cochain-finite**, i.e., all  $\lambda$ -cochains of  $A$  are finite. □

Given  $\lambda$ -CPOs  $A$  and  $B$ ,  $A \rightarrow_c B$  denotes the set of  $\lambda$ -continuous functions from  $A$  to  $B$ . Since  $\Omega$  and suprema of  $\lambda$ -chains of  $\lambda$ -continuous functions are  $\lambda$ -continuous,  $A \rightarrow_c B$  is a  $\lambda$ -CPO.

Let  $f : A \rightarrow A$ .

$a \in A$  is  **$f$ -closed** or  **$f$ -reductive** if  $f(a) \leq a$ .

$a$  is  **$f$ -dense** or  **$f$ -extensive** if  $a \leq f(a)$ .

$a$  is a **fixpoint** of  $f$  if  $f(a) = a$ .

**Kleene's Fixpoint Theorem** [84] (also known as Kleene's first recursion theorem)

(1) Let  $A$  be an  $\omega$ -CPO,  $f : A \rightarrow A$  be monotone and the **upper Kleene closure**  $f^\infty =_{\text{def}} \bigsqcup_{n < \omega} f^n(\perp)$  be  $f$ -closed. Then  $f^\infty$  is the least fixpoint of  $f$ .

(2) Let  $A$  be an  $\omega$ -co-CPO,  $f : A \rightarrow A$  be monotone and the **lower Kleene closure**  $f_\infty =_{\text{def}} \bigcap_{n < \omega} f^n(\top)$  be  $f$ -dense. Then  $f_\infty$  is the greatest fixpoint of  $f$ .

*Proof.* (1) Since  $f$  is monotone,  $\{f^n(\perp) \mid n < \omega\}$  is an  $\omega$ -chain.

Let  $a$  be  $f$ -closed. Then  $f^n(\perp) \leq a$  for all  $n \in \mathbb{N}$ . (3)

We show (3) by induction on  $n$ :  $f^0(\perp) = \perp \leq a$ . If  $f^n(\perp) \leq a$ , then  $f^{n+1}(\perp) \leq f(a) \leq a$  because  $f$  is monotone and  $a$  is  $f$ -closed.

Since  $f^\infty$  is  $f$ -closed,  $f(f^\infty)$  is  $f$ -closed. Hence by (3),  $f^\infty = \bigsqcup_{n < \omega} f^n(\perp) \leq f(f^\infty)$ , i.e.,  $f$  is also  $f$ -dense. We conclude that  $f^\infty$  is a fixpoint of  $f$ .

Let  $a$  be a fixpoint of  $f$ . Then  $a$  is  $f$ -closed and thus by (3),  $f^n(\perp) \leq a$  for all  $n \in \mathbb{N}$ . Hence  $f^\infty \leq a$ , i.e.,  $f^\infty$  is the least fixpoint of  $f$ .

(2) Analogously. □

**Proposition INF1** Let  $f : A \rightarrow A$  be monotone.

- (1) If  $f^\infty = f^n(\perp)$  for some  $n \in \mathbb{N}$ , then  $f^\infty$  is  $f$ -closed.
- (2) If  $f$  is  $\omega$ -continuous, then  $f^\infty$  is  $f$ -closed.

- (3) If  $f_\infty = f^n(\top)$  for some  $n \in \mathbb{N}$ , then  $f_\infty$  is  $f$ -dense.
- (4) If  $f$  is  $\omega$ -continuous, then  $f_\infty$  is  $f$ -dense.

*Proof.* (1) Suppose that  $f^\infty = f^n(\perp)$  holds true for some  $n \in \mathbb{N}$ . Then

$$f(f^\infty) = f(f^n(\perp)) = f^{n+1}(\perp) \leq \bigsqcup_{i<\omega} f^i(\perp) = f^\infty.$$

- (2) Suppose that  $f$  is  $\omega$ -continuous. Then

$$f(f^\infty) = f(\bigsqcup_{i<\omega} f^i(\perp)) \leq \bigsqcup_{i<\omega} f(f^i(\perp)) \leq \bigsqcup_{i<\omega} f^i(\perp) = f^\infty.$$

- (3) and (4): Analogously. □

**Proposition INF2** Let  $f : A \rightarrow A$  be monotone. Then for all  $n \in \mathbb{N}$ ,

$$f^\infty = f^n(\perp) \Leftrightarrow \forall i > n : f^i(\perp) = f^n(\perp), \quad (1)$$

$$f_\infty = f^n(\perp) \Leftrightarrow \forall i > n : f^i(\top) = f^n(\top). \quad (2)$$

*Proof.* (1) “ $\Rightarrow$ ”: Let  $f^\infty = f^n(\perp)$ . Then for all  $i > n$ ,  $f^i(\perp) \leq \bigsqcup_{k<\omega} f^k(\perp) = f^\infty = f^n(\perp)$  and thus  $f^i(\perp) = f^n(\perp)$  because  $f^n(\perp) \leq f^i(\perp)$ .

“ $\Leftarrow$ ”: Suppose that for all  $i > n$ ,  $f^i(\perp) = f^n(\perp)$ . Then for all  $i \in \mathbb{N}$ ,  $f^i(\perp) \leq f^n(\perp)$ , and thus  $f^\infty = \bigsqcup_{i<\omega} f^i(\perp) \leq f^n(\perp)$ . Hence  $f^\infty = f^n(\perp)$  because  $f^n(\perp) \leq \bigsqcup_{i<\omega} f^i(\perp) = f^\infty$ .

(2) Analogously. □

## Fixpoint Theorem for finite posets

Let  $A$  be a finite poset and  $f : A \rightarrow A$  be monotone.

- (1) If  $A$  has a least element  $\perp$ , then for some  $n < \omega$ ,  $f^\infty = f^n(\perp)$  is the least fixpoint of  $f$ .
- (2) If  $A$  has a greatest element  $\top$ , then for some  $n < \omega$ ,  $f_\infty = f^n(\top)$  is the greatest fixpoint of  $f$ .

*Proof.* (1) Since  $f$  is monotone, induction on  $i$  implies  $f^i(\perp) \leq f^{i+1}(\perp)$  for all  $i \in \mathbb{N}$ . Hence there is  $n \in \mathbb{N}$  such that  $f^n(\perp) = f^{n+1}(\perp)$  because  $A$  is finite. Therefore,  $f(f^n(\perp)) = f^{n+1}(\perp) = f^n(\perp) \leq f(f^n(\perp))$  and thus  $f(f^n(\perp)) = f^n(\perp)$ . Induction on  $i$  implies  $f^i(\perp) = f^n(\perp)$  for all  $i > n$ . Hence by Proposition INF2 (1),  $f^\infty = f^n(\perp)$ , and thus by Proposition INF1 (1),  $f^\infty$  is  $f$ -closed. We conclude by Kleene's Fixpoint Theorem that  $f^\infty$  is the least fixpoint of  $f$ .

(2) Analogously. □

Hence, if  $A$  is finite, then the function

$$\begin{aligned} \text{fixpt} : \mathcal{P}(A \times A) &\rightarrow (A \rightarrow A) \rightarrow A \rightarrow A \\ (\leq) &\rightarrow \lambda f. \lambda a. \text{if } f(a) \leq a \text{ then } a \text{ else } \text{fixpt}(\leq)(f)(f(a)) \end{aligned}$$

computes least and greatest fixpoints:

$$f^\infty = \bigsqcup_{i<\omega} f^i(\perp) = \text{fixpt}(\leq)(f)(\perp), \quad f_\infty = \prod_{i<\omega} f^i(\top) = \text{fixpt}(\geq)(f)(\top).$$

The Fixpoint Theorem for finite posets can be generalized from ordinal  $\omega$  to any ordinal  $\lambda$ :

**Zermelo's Fixpoint Theorem** ([2], Prop. 1.3.1; [93], Extended Folk Theorem 6; [10], Thm. 4.1.1)

(1) Let  $A$  be a  $\lambda$ -CPO with  $|A| < \lambda$ ,  $f : A \rightarrow A$  be monotone and  $B = \{f^i(\perp) \mid i < \lambda\}$  be the  $\lambda$ -chain of  $A$  that is defined as follows:  $f^0(\perp) = \perp$ , for all successor ordinals  $i + 1 < \lambda$ ,  $f^{i+1}(\perp) = f(f^i(\perp))$ , and for all limit ordinals  $i < \lambda$ ,  $f^i(\perp) = \bigsqcup_{k \in i} f^k(\perp)$ .  $f^{|A|}(\perp)$  is the least fixpoint of  $f$ .

(2) Let  $A$  be a  $\lambda$ -co-CPO with  $|A| < \lambda$ ,  $f : A \rightarrow A$  be monotone and  $B = \{f^i(\top) \mid i < \lambda\}$  be the  $\lambda$ -cochain of  $A$  that is defined as follows:  $f^0(\top) = \top$ , for all successor ordinals  $i + 1 < \lambda$ ,  $f^{i+1}(\top) = f(f^i(\top))$ , and for all limit ordinals  $i < \lambda$ ,  $f^i(\top) = \prod_{k \in i} f^k(\top)$ .

$f^{|A|}(\top)$  is the greatest fixpoint of  $f$ .

*Proof.* (1) First we show by transfinite induction on  $i$  that for all  $i < \lambda$ ,

$$f^i(\perp) \text{ is defined and for all } k \leq i, f^k(\perp) \leq f^i(\perp). \quad (3)$$

Of course,  $f^0(\perp) = \perp$  is defined. Let  $i+1 < \lambda$  be a successor ordinal. Then by induction hypothesis,  $f^i(\perp)$  is defined and for all  $k \leq i$ ,  $f^k(\perp) \leq f^i(\perp)$ . Hence  $f^{i+1}(\perp) = f(f^i(\perp))$  is defined. Since  $f$  is monotone, for all  $k \leq i$ ,  $f^{k+1}(\perp) = f(f^k(\perp)) \leq f(f^i(\perp)) = f^{i+1}(\perp)$ , and thus for all  $k \leq i+1$ ,  $f^{k+1}(\perp) \leq f^{i+1}(\perp)$ .

Let  $i$  be a limit ordinal. Then by induction hypothesis, for all  $k \in i$ ,  $f^k(\perp)$  is defined and for all  $j \leq k$ ,  $f^j(\perp) \leq f^k(\perp)$ . Hence  $C = \{f^k(\perp) \mid k \in i\}$  is a  $\lambda$ -chain and thus  $f^i(\perp) = \bigsqcup C$  exists. Hence for all  $k \in i$ ,  $f^k(\perp) \leq f^i(\perp)$ .

We conclude from (3) that  $B$  is a  $\lambda$ -chain.

Assume that  $f^{|A|}(\perp) \neq f(f^{|A|}(\perp))$ . Then for all  $i \leq |A| + 1$ ,  $f^i(\perp) < f(f^i(\perp))$ , and thus we obtain the contradiction  $|\{f^i(\perp) \mid i \leq |A| + 1\}| > |A|$ .

Let  $b$  be a fixpoint of  $f$ . We show by transfinite induction on  $i$  that for all  $i < \lambda$

$$f^i(\perp) \leq b. \quad (4)$$

Of course,  $f^0(\perp) = \perp \leq b$ . Let  $i+1 > \lambda$  be a successor ordinal. Then by induction hypothesis,  $f^i(\perp) \leq b$  and thus  $f^{i+1}(\perp) = f(f^i(\perp)) \leq f(b) = b$  because  $f$  is monotone.

Let  $i$  be a limit ordinal. Then  $f^i(\perp) = \bigsqcup\{f^k(\perp) \mid k \in i\}$ . By induction hypothesis, for all  $k \in i$ ,  $f^k(\perp) \leq b$ . Hence  $f^i(\perp) \leq b$ .

We conclude from (4) that  $f^{|A|}(\perp)$  is the *least* fixpoint of  $f$ .

(2) Analogously. □

A poset  $A$  is a **complete lattice** if each subset  $B$  of  $A$  has a supremum  $\bigsqcup B$  and an infimum  $\bigsqcap B$  in  $A$ .

Then  $\bigsqcap B = \bigsqcup\{a \in A \mid \forall b \in B : a \leq b\}$  is the infimum of  $B$ ,  $\perp = \bigsqcup \emptyset = \bigsqcap A$  is the least element and  $\top = \bigsqcup A = \bigsqcap \emptyset$  is the greatest element of  $A$ .

Let  $A, B$  be complete lattices.

$f : A \rightarrow B$  is **continuous** if for all  $C \subseteq A$ ,  $f(\bigsqcup C) = \bigsqcup_{a \in C} f(a)$ .

$f : A \rightarrow B$  is **cocontinuous** if for all  $C \subseteq A$ ,  $f(\bigsqcap C) = \bigsqcap_{a \in C} f(a)$ .

**Proposition** If  $f$  is continuous or cocontinuous, then  $f$  is monotone.

*Proof.* Let  $a \leq b$ . Then  $a \sqcap b = a$  and  $a \sqcup b = b$  and thus  $f(a) \sqcap f(b) = f(a \sqcap b) = f(a)$  or  $f(a) \sqcup f(b) = f(a \sqcup b) = f(b)$ . Hence  $f(a) \leq f(b)$ .  $\square$

**Proposition** Let  $f$  be monotone.

$f$  is continuous iff for all  $C \subseteq A$ ,  $f(\bigsqcup C) \leq \bigsqcup_{a \in C} f(a)$ .

$f$  is cocontinuous iff for all  $C \subseteq A$ ,  $\bigsqcap_{a \in C} f(a) \leq f(\bigsqcap C)$ .  $\square$

## Fixpoint Theorem of Knaster and Tarski [161]

Let  $A$  be a complete lattice and  $f : A \rightarrow A$  be monotone.

- (1)  $\text{lfp}(f) =_{\text{def}} \bigsqcap \{a \in A \mid a \text{ is } f\text{-closed}\}$  is the least fixpoint of  $f$ .
- (2)  $f^\infty \leq \text{lfp}(f)$ .
- (3) If  $f^\infty$  is  $f$ -closed, then  $\text{lfp}(f) \leq f^\infty$ .
- (4) If  $f^\infty$  is  $f$ -closed, then  $\text{lfp}(f)$  is the least fixpoint of  $f$ .
- (5)  $\text{gfp}(f) =_{\text{def}} \bigsqcup \{a \in A \mid a \text{ is } f\text{-dense}\}$  is the greatest fixpoint of  $f$ .

- (6)  $gfp(f) \leq f_\infty$ .
- (7) If  $f_\infty$  is  $f$ -dense, then  $f_\infty \leq gfp(f)$ .
- (8) If  $f_\infty$  is  $f$ -dense, then  $gfp(f)$  is the greatest fixpoint of  $f$ .

*Proof.*

- (1) Let  $a$  be  $f$ -closed. Then  $lfp(f) \leq a$  and thus  $f(lfp(f)) \leq f(a) \leq a$  because  $f$  is monotone, i.e.,  $f(lfp(f))$  is a lower bound of all  $f$ -closed elements of  $A$ .

Hence (9)  $f(lfp(f)) \leq \bigcap\{a \in A \mid a \text{ is } f\text{-closed}\} = lfp(f)$ , i.e.,  $f(lfp(f))$  is  $f$ -closed, and thus (10)  $lfp(f) = \bigcap\{a \in A \mid a \text{ is } f\text{-closed}\} \leq f(lfp(f))$ . By (9) and (10),  $lfp(f)$  is a fixpoint of  $f$ .

Let  $a$  be a fixpoint of  $f$ . Then  $a$  is  $f$ -closed and thus  $lfp(f) \leq a$ , i.e.,  $lfp(f)$  is the *least* fixpoint of  $f$ .

- (2) By induction on  $n$ , we obtain  $f^n(\perp) \leq lfp(f)$ :  $f^0(\perp) = \perp \leq lfp(f)$  and

$$f^{n+1}(\perp) = f(f^n(\perp)) \stackrel{\text{ind. hyp.}}{\leq} f(lfp(f)) \stackrel{(1)}{=} lfp(f)$$

because  $f$  is monotone. Hence  $f^\infty = \bigsqcup_{n < \omega} f^n(\perp) \leq lfp(f)$ .

- (3) Let  $f^\infty$  be  $f$ -closed. Then  $lfp(f) = \bigcap\{a \in A \mid a \text{ is } f\text{-closed}\} \leq f^\infty$ .

(4) follows directly from (1)-(3).

(5)-(8) can be proved analogously. □

Compared with Kleene's Fixpoint Theorem, the Fixpoint Theorem of Knaster and Tarski only requires monotonicity of  $f$ , but provides non-constructive fixpoints of  $f$ .

## Fixpoint induction

Let  $f : A \rightarrow A$  be monotone, called a **step function**. Suppose that

- (a)  $A$  is a complete lattice or a  $\lambda$ -CPO with  $|A| < \lambda$ , or
- (b)  $A$  is an  $\omega$ -CPO and  $f$  is  $\omega$ -continuous.

- (1) For all  $f$ -closed  $a \in A$ ,  $\text{lfp}(f) \leq a$ .
- (2) For all  $n > 0$  and  $f^n$ -closed  $a \in A$ ,  $\text{lfp}(f) \leq a$ .

*Proof.* (1) Let (a) hold true. If  $A$  is a complete lattice, then by the Fixpoint Theorem of Knaster and Tarski,  $\text{lfp}(f) = \bigcap\{a \in A \mid f(a) \leq a\} \leq a$ . If  $A$  is a  $\lambda$ -CPO, then by transfinite induction on  $i$ , for all  $i < \lambda$ ,  $f^i(\perp) \leq a$  because  $f$  is monotone and  $a$  is  $f$ -closed. Hence by Zermelo's Fixpoint Theorem,  $\text{lfp}(f) = f^{|A|}(\perp) \leq a$ .

Let (b) hold true. By induction on  $n$ , for all  $i \in \mathbb{N}$ ,  $f^i(\perp) \leq a$  because  $f$  is monotone and  $a$  is  $f$ -closed.

Hence by Kleene's Fixpoint Theorem (1),  $\text{lfp}(f) = \bigsqcup_{i<\omega} f^i(\perp) \leq a$ .

(2) Let (a) hold true. If  $A$  is a complete lattice, then

$$b =_{\text{def}} \prod_{i>0} f^i(a) \leq f^n(a) \leq a = f^0(a). \quad (4)$$

By (3), for all  $i > 0$ ,  $b \leq f^{i-1}(a)$  and thus  $f(b) \leq f^i(a)$  because  $f$  is monotone. Hence  $f(b)$  is a lower bound of  $\{f^i(a) \mid i > 0\}$  and thus  $f(b) \leq b$ , i.e.,  $b$  is  $f$ -closed. By the Fixpoint Theorem of Knaster and Tarski,  $\text{lfp}(f) = \prod\{c \in A \mid f(c) \leq c\}$ . Hence (3) implies  $\text{lfp}(f) \leq b \leq a$ . If  $A$  is a  $\lambda$ -CPO, then by transfinite induction on  $i$ , for all  $i < \lambda$ ,  $f^{n*i}(\perp) \leq a$  because  $f$  is monotone and  $a$  is  $f$ -closed. Hence by Zermelo's Fixpoint Theorem,  $\text{lfp}(f) = f^{|A|}(\perp) \leq a$ .

Let (b) hold true. By induction on  $i$ , for all  $i \in \mathbb{N}$ ,  $f^{n*i}(\perp) \leq a$  because  $f$  is monotone and  $a$  is  $f$ -closed. Hence by Kleene's Fixpoint Theorem (1),  $\text{lfp}(f) = \bigsqcup_{i<\omega} f^i(\perp) = \bigsqcup_{i<\omega} f^{n*i}(\perp) \leq a$ .  $\square$

## Fixpoint coinduction

Let  $f : A \rightarrow A$  be monotone, called a **step function**. Suppose that

- (a)  $A$  is a complete lattice or a  $\lambda$ -co-CPO with  $|A| < \lambda$ , or
  - (b)  $A$  is an  $\omega$ -co-CPO and  $f$  is  $\omega$ -cocontinuous.
- (1) For all  $f$ -dense  $a \in A$ ,  $a \leq gfp(f)$ .
  - (2) For all  $n > 0$  and  $f^n$ -dense  $a \in A$ ,  $a \leq gfp(f)$ .

*Proof.* Analogously. □

## Computational induction

Let  $A$  be an  $\omega$ -CPO,  $f : A \rightarrow A$  be  $\omega$ -continuous and  $B$  be an **admissible** subset of  $A$ , i.e., for all  $\omega$ -chains  $C$  of  $A$ ,  $C \subseteq B$  implies  $\bigsqcup C \in B$ .

If  $\perp \in B$  and for all  $b \in B$ ,  $f(b) \in B$ , then  $lfp(f) \in B$ . (1)

*Proof.* (1) provides the induction base and the induction step of a proof by induction on  $n$  that for all  $n \in \mathbb{N}$ ,  $f^n(\perp) \in B$ . Since  $B$  is admissible, we conclude  $lfp(f) = \bigsqcup_{n<\omega} f^n(\perp) \in B$  by Kleene's Fixpoint Theorem (1). □

## Computational coinduction

Let  $A$  be an  $\omega$ -co-CPO,  $f : A \rightarrow A$  be  $\omega$ -cocontinuous and  $B$  be an **co-admissible** subset of  $A$ , i.e., for all  $\omega$ -cochains  $C$  of  $A$ ,  $C \subseteq B$  implies  $\prod C \in B$ .

If  $\top \in B$  and for all  $b \in B$ ,  $f(b) \in B$ , then  $gfp(f) \in B$ .

*Proof.* Analogously. □

## Noetherian induction

Let  $A$  be a class,  $R$  be a well-founded relation on  $A$  and  $B$  be a subset of  $A$ . Suppose that

$$\text{for all } a \in A, (\forall b \in A : bRa \Rightarrow b \in B) \text{ implies } a \in B. \quad (1)$$

Then  $B = A$ .

*Proof.* Suppose that (1) holds true, but there is  $a \in A \setminus B$ . (1) implies  $bRa$  and  $b \notin B$  for some  $b \in A$ , i.e.,  $b \in A \setminus B$ . We may repeat this conclusion (with  $b$  instead of  $a$ ) infinitely often and thus obtain a subset of  $A$  that has no least element w.r.t.  $R$ . □

If  $R$  is a well-order, then Noetherian induction is also called **transfinite induction**.

## Categories

<i>poset notion</i>	<i>categorical notion</i>
p(artially) o(rdered) set	category
$A$	$\mathcal{K}$
element	object
$a \in A$	$A$
ordered pair	morphism
$a \leq b$	$f : A \rightarrow B$
least element	initial object
greatest element	final object
subset	diagram
$S \subseteq A$	A $\mathcal{K}$ -diagram $D$ is a directed graph $G = (N, E, \text{source}, \text{target} : E \rightarrow N)$ with node labels in $\mathcal{K}$ and edge labels in $Mor(\mathcal{K})$

upper bound of  $S$   
lower bound of  $S$

supremum (least upper bound) of  $S$   
infimum (greatest lower bound) of  $S$

$\mathcal{S} \subseteq \mathcal{P}(A)$

$A$  is  $\mathcal{S}$ -complete:  
each  $S \in \mathcal{S}$  has supremum  $\sqcup S$

$A$  is  $\mathcal{S}$ -cocomplete:  
each  $S \in \mathcal{S}$  has infimum  $\sqcap S$

$A$  is a complete lattice:  
all subsets of  $\mathcal{S}$  have suprema and infima

cocone of  $D$   
cone of  $D$

colimit of  $D$   
limit of  $D$

$\mathcal{S}(G)$  = class of all  $\mathcal{K}$ -diagrams  
with underlying graph  $G$

$\mathcal{K}$  is  $\mathcal{S}(G)$ -cocomplete  
each  $D \in \mathcal{S}(G)$  has colimit  $\text{col}(D)$

$\mathcal{K}$  is  $\mathcal{S}(G)$ -complete  
each  $D \in \mathcal{S}(G)$  has limit  $\text{lim}(D)$

$\mathcal{K}$  is a complete and cocomplete category  
all  $\mathcal{K}$ -diagrams have limits and colimits

monotone function  $f : A \rightarrow B$

$$a \leq b \Rightarrow f(a) \leq f(b)$$

$f$ -closed element  $a$ :  $f(a) \leq a$

$f$ -dense element  $a$ :  $a \leq f(a)$

Knaster-Tarski Fixpoint Theorem

$f : A \rightarrow B$  is monotone  $\Rightarrow$

$\prod\{a \in A \mid f(a) \leq a\}$  is least fixpoint of  $f$

$f : A \rightarrow B$  is monotone  $\Rightarrow$

$\bigsqcup\{a \in A \mid a \leq f(a)\}$  is greatest fixpoint of  $f$

$f : A \rightarrow B$  is  $\mathcal{S}$ -continuous:

$$\forall S \in \mathcal{S} : f(\bigsqcup S) = \bigsqcup f(S)$$

$f : A \rightarrow B$  is  $\mathcal{S}$ -cocontinuous:

$$\forall S \in \mathcal{S} : f(\prod S) = \prod f(S)$$

functor  $F : \mathcal{K} \rightarrow \mathcal{L}$

$$A \xrightarrow{f} B \Rightarrow F(A) \xrightarrow{F(f)} F(B)$$

$\alpha$   $F$ -algebra:  $F(A) \xrightarrow{\alpha} A$

$\alpha$   $F$ -coalgebra:  $A \xrightarrow{\alpha} F(A)$

Lambek's Lemma

$\alpha : F(A) \rightarrow A$  is initial  $F$ -algebra

$$\Rightarrow A \text{ is fixpoint of } F : \mathcal{K} \rightarrow \mathcal{K} \quad (1)$$

$\alpha : A \rightarrow F(A)$  is final  $F$ -coalgebra

$$\Rightarrow A \text{ is fixpoint of } F : \mathcal{K} \rightarrow \mathcal{K} \quad (2)$$

$F : \mathcal{K} \rightarrow \mathcal{L}$  is  $\mathcal{S}(G)$ -continuous:

$$\forall D \in \mathcal{S}(G) : F(\text{col}(D)) = \text{col}(F(D))$$

$F : \mathcal{K} \rightarrow \mathcal{L}$  is  $\mathcal{S}(G)$ -cocontinuous:

$$\forall D \in \mathcal{S}(G) : F(\text{lim}(D)) = \text{lim}(F(D))$$

Kleene's Fixpoint Theorem

$$\mathcal{S} = \{(a_n)_{n \in \mathbb{N}} \mid \forall n \in \mathbb{N} : a_n \leq a_{n+1}\} \quad G = (\mathbb{N}, \{(n, n+1) \mid n \in \mathbb{N}\})$$

$f : A \rightarrow B$  is  $\mathcal{S}$ -continuous

$\Rightarrow \bigsqcup_{i \in \omega} f^i(\perp)$   
is least fixpoint of  $f$

Adamek-Koubek Fixpoint Theorem

$F : \mathcal{K} \rightarrow \mathcal{L}$  is  $\mathcal{S}(G)$ -continuous,  
 $D = (F^n(Ini) \rightarrow F^{n+1}(Ini))_{n \in \mathbb{N}}$   
 $\Rightarrow F(col(D)) \rightarrow col(D)$   
 is initial  $F$ -algebra  
 and thus, by (1), fixpoint of  $F$

$$\mathcal{S} = \{(a_n)_{n \in \mathbb{N}} \mid \forall n \in \mathbb{N} : a_n \geq a_{n+1}\} \quad G = (\mathbb{N}, \{(n+1, n) \mid n \in \mathbb{N}\})$$

$f : A \rightarrow B$  is  $\mathcal{S}$ -cocontinuous

$\Rightarrow \prod_{i \in \omega} f^i(\top)$   
is greatest fixpoint of  $f$

$F : \mathcal{K} \rightarrow \mathcal{L}$  is  $\mathcal{S}(G)$ -cocontinuous,  
 $D = (F^{n+1}(Fin) \rightarrow F^n(Fin))_{n \in \mathbb{N}}$   
 $\Rightarrow lim(D) \rightarrow F(lim(D))$   
 is final  $F$ -coalgebra  
 and thus, by (2), fixpoint of  $F$

Galois connection

adjunction

$$(f : A \rightarrow B, g : B \rightarrow A) \quad F : \mathcal{K} \rightarrow \mathcal{L} \dashv G : \mathcal{L} \rightarrow \mathcal{K}$$

$$f(a) \leq b \Leftrightarrow a \leq g(b) \quad \frac{A \rightarrow G(B)}{F(A) \rightarrow B}$$

A **category**  $\mathcal{K}$  consists of

- a class of  **$\mathcal{K}$ -objects**, also denoted by  $\mathcal{K}$ ,
- for all  $A, B \in \mathcal{K}$  a set  $\mathcal{K}(A, B)$  of  **$\mathcal{K}$ -morphisms**, also called **arrows**,
- for all  $A, B, C \in \mathcal{K}$  a function

$$\circ : \mathcal{K}(B, C) \times \mathcal{K}(A, B) \rightarrow \mathcal{K}(A, C),$$

called **composition**, such that for all  $A, B, C, D \in \mathcal{K}$ ,  $f \in \mathcal{K}(A, B)$ ,  $g \in \mathcal{K}(B, C)$  and  $h \in \mathcal{K}(C, D)$ ,

$$h \circ (g \circ f) = (h \circ g) \circ f.$$

- for all  $A \in \mathcal{K}$  an **identity**  $id_A \in \mathcal{K}(A, A)$  such that for all  $B \in \mathcal{K}$  and  $f \in \mathcal{K}(A, B)$ ,

$$f \circ id_A = f = id_B \circ f.$$

**$Mor(\mathcal{K})$**  denotes the class of all sets  $\mathcal{K}(A, B)$  with  $A, B \in \mathcal{K}$ .

$f \in \mathcal{K}(A, B)$  is usually written as  $f : A \rightarrow B$ .  $A$  and  $B$  are called the **source** and **target** of  $f$ , respectively.

$\mathcal{K}$  is **small** if the class of all objects of  $\mathcal{K}$  is a set.

A category  $\mathcal{L}$  is a **subcategory** of  $\mathcal{K}$  if all objects of  $\mathcal{L}$  are objects of  $\mathcal{K}$  and all  $\mathcal{L}$ -morphisms are  $\mathcal{K}$ -morphisms.  $\mathcal{L}$  is **full** if all  $\mathcal{K}$ -morphisms between objects of  $\mathcal{L}$  are  $\mathcal{L}$ -morphisms.

## Examples

$\text{Set}$  and  $\text{Set}_{\neq\emptyset}$  denote the categories of sets or nonempty sets as objects and functions as morphisms. In terms of [160], section 2.1.1, a set  $X$  can be thought of as a collection of things each of which is recognizable as being in  $X$  and such that for each two elements of  $X$  we can tell whether they are equal or not.

$\text{Pfn}$  denotes the category of sets as objects and *partial functions* as morphisms, i.e., for all sets  $A, B$ ,

$$\text{Pfn}(A, B) = (A \rightarrow B + 1)$$

(see, e.g., [97], section 1.3). Composition and identities are defined as follows: For all sets  $A, B, C$ ,

$$\begin{aligned} \circ^{\text{Pfn}} : \text{Pfn}(B, C) \times \text{Pfn}(A, B) &\rightarrow \text{Pfn}(A, C) \\ (g, f) &\mapsto \lambda a. \text{if } f(a) = () \text{ then } () \text{ else } g(f(a)), \end{aligned}$$

$$id_A^{Pfn} = \lambda a.a.$$

In terms of chapter 23,  $\circ^{Pfn}$  is Kleisli composition and  $id^{Pfn}$  is the unit of the monad  $\_ + 1$ .

**Rel** denotes the category of sets as objects and binary relations as morphisms, i.e., for all sets  $A, B$ ,

$$Rel(A, B) = \mathcal{P}(A \times B).$$

Composition and identities are defined as follows: For all sets  $A, B, C$ ,

$$\begin{aligned} \circ^{Rel} : Rel(B, C) \times Rel(A, B) &\rightarrow Rel(A, C) \\ (r', r) &\mapsto \{(a, c) \in A \times C \mid \exists b \in B : (a, b) \in r \wedge (b, c) \in r'\}, \end{aligned}$$

$$id_A^{Rel} = \Delta_A^2.$$

**Mfn** denotes the category of sets as objects and **multivalued** or **nondeterministic** functions as morphisms, i.e., for all sets  $A, B$ ,

$$Mfn(A, B) = (A \rightarrow \mathcal{P}(B))$$

(see, e.g., [97], section 1.4).

Composition and identities are defined as follows: For all sets  $A, B, C$ ,

$$\circ^{\textcolor{blue}{Mfn}} : Mfn(B, C) \times Mfn(A, B) \rightarrow Mfn(A, C)$$

$$(g, f) \mapsto \lambda a. \{c \in C \mid \exists b \in f(a) : c \in g(b)\}.$$

$$id_A^{\textcolor{blue}{Mfn}} = \lambda a. \{a\}.$$

In terms of chapter 23,  $\circ^{\textcolor{blue}{Mfn}}$  is Kleisli composition and  $id^{\textcolor{blue}{Mfn}}$  is the unit of the powerset monad.

**Exercise 1** Show that  $Pfn$ ,  $Rel$  and  $Mfn$  are categories. □

$f \in \mathcal{K}(A, B)$  is (an) **epi(morphism)** if for all  $g, h \in \mathcal{K}(B, C)$ ,  $g \circ f = h \circ f$  implies  $g = h$ .

$f \in \mathcal{K}(A, B)$  is (a) **mono(morphism)** if for all  $g, h \in \mathcal{K}(C, A)$ ,  $f \circ g = f \circ h$  implies  $g = h$ .

$f \in \mathcal{K}(A, B)$  is a **retraction** or **split epi** if  $f \circ g = id_B$  for some  $g \in \mathcal{K}(B, A)$ .

$f \in \mathcal{K}(A, B)$  is a **coretraction**, **section** or **split mono** if  $g \circ f = id_A$  for some  $g \in \mathcal{K}(B, A)$ .

Exercise 2 Show that retractions are epi and coretractions are mono. □

Exercise 3 Show that a function is surjective (injective) iff it is epi (mono) in *Set*. □

$f \in \mathcal{K}(A, B)$  is (an) **iso(morphism)** and  $A$  and  $B$  are **isomorphic**, written as  $A \cong B$ , if  $f$  is a retraction and a coretraction. Two isomorphic objects are often regarded as a single one, in particular, if they have the same categorical (“universal”) properties.

$f \in \mathcal{K}(A, B)$  is an **embedding** and  $A$  is **embedded** in  $B$  if  $f$  is mono.

Exercise 4 Show that for every iso  $f \in \mathcal{K}(A, B)$  there is exactly one  $g \in \mathcal{K}(B, A)$  with  $g \circ f = id_A$  and  $f \circ g = id_B$ . This justifies the notation  $f^{-1}$  for  $g$  and the phrasing: “ $g$  is *the inverse of  $f$* ”. □

**Lemma EPIMON** Let  $f \in \mathcal{K}(A, B)$  and  $g \in \mathcal{K}(B, C)$ .

- (1) If  $g \circ f$  is epi, then  $g$  is epi.
- (2) If  $g \circ f$  is mono, then  $f$  is mono. □

**Lemma ISO** Let  $f \in \mathcal{K}(A, C)$ ,  $g \in \mathcal{K}(A, B)$  and  $h \in \mathcal{K}(B, C)$ .

(1) If  $g$  is iso, then

$$f = h \circ g \iff h = f \circ g^{-1}.$$

(2) If  $h$  is iso, then

$$f = h \circ g \iff g = h^{-1} \circ f. \quad \square$$

The **dual category**  $\mathcal{K}^{op}$  of  $\mathcal{K}$  is constructed from  $\mathcal{K}$  by keeping the objects, but reversing the arrows, i.e., for all  $A, B \in \mathcal{K}$ ,  $\mathcal{K}^{op}(A, B) = \mathcal{K}(B, A)$ .

The formulation of a property  $\varphi$  of  $\mathcal{K}$  as a property  $\psi$  of  $\mathcal{K}^{op}$  is called **dualization**. The **dual property**  $\psi$  is obtained from  $\varphi$  by reversing all arrows mentioned in  $\varphi$ .

Let  $I$  be a set (of indices) and  $\mathcal{K}_i$ ,  $i \in I$ , be categories.

The **product category**  $\prod_{i \in I} \mathcal{K}_i$  has tuples  $(A_i)_{i \in I}$  with  $A_i \in \mathcal{K}_i$  for all  $i \in I$  as objects and tuples  $(f_i)_{i \in I}$  with  $f_i \in \mathcal{K}_i(A_i, B_i)$  for all  $i \in I$  as morphisms.

If there is a category  $\mathcal{K}$  such that for all  $i \in I$ ,  $\mathcal{K}_i = \mathcal{K}$ , then  $\prod_{i \in I} \mathcal{K}_i$  is also written as  $\mathcal{K}^I$  and called a **power category**.

A  $\mathcal{K}$ -object  $A$  is **initial** in  $\mathcal{K}$  if for all  $\mathcal{K}$ -objects  $B$  there is a unique  $\mathcal{K}$ -morphism  $ini^B : A \rightarrow B$ .

A  $\mathcal{K}$ -object  $A$  is **final** or **terminal** in  $\mathcal{K}$  if for all  $\mathcal{K}$ -objects  $B$  there is a unique  $\mathcal{K}$ -morphism  $fin^B : B \rightarrow A$ .

All initial  $\mathcal{K}$ -objects are isomorphic. Every  $\mathcal{K}$ -object that is isomorphic to an initial one is initial.

All final  $\mathcal{K}$ -objects are isomorphic. Every  $\mathcal{K}$ -object that is isomorphic to a final one is final.

### Examples:

The empty set is initial in  $Set$ . There are no initial objects in  $Set_{\neq \emptyset}$ . Every one-element set is final in  $Set$  and  $Set_{\neq \emptyset}$ .

### Lemma MINMAX

- (1) Let  $A$  be initial in  $\mathcal{K}$ . All  $\mathcal{K}$ -monomorphisms  $f : B \rightarrow A$  are isomorphisms.
- (2) Let  $A$  be final in  $\mathcal{K}$ . All  $\mathcal{K}$ -epimorphisms  $g : A \rightarrow B$  are isomorphisms.

*Proof.*

- (1) Since  $A$  is initial in  $\mathcal{K}$ ,  $f \circ ini^B = id_A$ . Hence  $f \circ ini^B \circ f = id_A \circ f = f = f \circ id_B$  and thus  $ini^B \circ f = id_B$  because  $f$  is mono. Hence  $f$  is iso.
- (2) Since  $A$  is final in  $\mathcal{K}$ ,  $fin^B \circ g = id_A$ . Hence  $g \circ fin^B \circ g = g \circ id_A = g = id_B \circ g$  and thus  $g \circ fin^B = id_B$  because  $g$  is epi. Hence  $g$  is iso.  $\square$

Let  $\mathcal{K}$  be a category with final object  $1_{\mathcal{K}}$ .  $X \in \mathcal{K}$  has the fixpoint property if for all  $\mathcal{K}$ -morphisms  $f : X \rightarrow X$  there is  $x : 1_{\mathcal{K}} \rightarrow X$  with  $f \circ x = x$ .

A  $\mathcal{K}$ -morphism  $f : A \rightarrow X$  is ubiquitous if for all  $\mathcal{K}$ -morphisms  $g : A \rightarrow X$  there is  $a : 1_{\mathcal{K}} \rightarrow A$  with  $f \circ a = g \circ a$ .

**Lawvere's Fixpoint Theorem** ([157], Thms. 1 quarto and 5; [178], Thm. 1)

Let  $\mathcal{K}$  be a category with final object  $1_{\mathcal{K}}$ .

- (1)  $X \in \mathcal{K}$  has the fixpoint property iff there is a ubiquitous  $\mathcal{K}$ -morphism  $f : A \rightarrow X$ .
- (2) Let  $\mathcal{K}$  be Cartesian closed (see [Adjunctions](#), Ex. 8) and  $f : A \rightarrow X^A$  be a **surjective morphism**, i.e., for all  $g : 1_{\mathcal{K}} \rightarrow X^A$  there is  $a_g : 1_{\mathcal{K}} \rightarrow A$  such that  $f \circ a_g = g$ . Then  $X$  has the fixpoint property.

*Proof.*

(1) Let  $f : A \rightarrow X$  be ubiquitous and  $g : X \rightarrow X$  be a  $\mathcal{K}$ -morphism. Then  $f \circ a_g = g \circ f \circ a_g$  for some  $a_g : 1_{\mathcal{K}} \rightarrow A$ , i.e.,  $f \circ a_g$  is a fixpoint of  $g$ . Conversely, suppose that  $X \in \mathcal{K}$  has the fixpoint property. Let  $g : X \rightarrow X$  be a  $\mathcal{K}$ -morphism with fixpoint  $x_g$ .

Then  $id_X(x_g) = x_g = g(x_g)$ . Hence the identity on  $X$  is ubiquitous.

(2) By (1), it is sufficient to find a ubiquitous  $\mathcal{K}$ -morphism  $h : A \rightarrow X$ . Define  $h$  as  $f^* \circ \langle id_A, id_A \rangle$  and let  $g : A \rightarrow X$ . Then

$$h \circ a_g = f^* \circ \langle id_A, id_A \rangle \circ a_g = f^* \circ \langle a_g, a_g \rangle = f \circ \pi_1 \circ \langle a_g, a_g \rangle = f \circ a_g.$$

Hence  $h$  is ubiquitous. □

## Corollaries

(1) Cantor: The set  $2^{\mathbb{N}}$  of infinite bit streams is uncountable.

*Proof.* Let  $\mathcal{K} = Set$ .  $g : 2 \rightarrow 2$  with  $g(0) = 1$  and  $g(1) = 0$  does not have a fixpoint. Hence by Lawvere's Fixpoint Theorem (2), there is no surjective morphism  $f : \mathbb{N} \rightarrow 2^{\mathbb{N}}$  and thus  $2^{\mathbb{N}}$  is uncountable.

(2) For all sets  $A$ ,  $|A| < |2^A|$ .

*Proof.* Same argument as in the proof of (1).

(3) Let  $\mathcal{K}$  be the category of classes,  $A$  be the class of all sets and  $\chi : A \rightarrow 2^A$  be the function that maps each subclass  $B$  of  $A$  to its indicator function  $\chi(B) : A \rightarrow 2$ .

$\chi$  is not surjective. In particular, the class of all sets  $S$  that do not contain  $S$  is not a set.

*Proof.* Let  $g : 2 \rightarrow 2$  with  $g(0) = 1$  and  $g(1) = 0$ . Assume that  $\chi$  is surjective. Then there would be a set  $T$  such that  $\chi(T) = \lambda S.g(\chi(S)(S))$ . Hence  $g(\chi(T)(T)) = \chi(T)(T)$  and thus  $f(T)(T)$  would be a fixpoint of  $g$ .  $\nabla$

Since for all sets  $S$ ,

$$g(\chi(S)(S)) = 1 \Leftrightarrow \chi(S)(S) = 0 \Leftrightarrow S \notin S,$$

but  $\lambda S.g(\chi(S)(S))$  does not have a pre-image under  $\chi$ , the class of all sets  $S$  that do not contain  $S$  is not a set.  $\square$

[157], section 3.1, and [178], §3 and §5, employ the same line of argument for re-establishing well-known “negative” results, such as the unsolvability of the halting problem (Turing), the incompleteness of arithmetic theories (Gödel) or the undefinability of truth (Tarski).

## Functors and natural transformations

Let  $\mathcal{K}$  and  $\mathcal{L}$  be two categories. A (**covariant**) **functor**  $F : \mathcal{K} \rightarrow \mathcal{L}$  maps each  $\mathcal{K}$ -object to an  $\mathcal{L}$ -object and each  $\mathcal{K}$ -morphism  $f : A \rightarrow B$  to an  $\mathcal{L}$ -morphism  $F(f) : F(A) \rightarrow F(B)$  such that

- for all  $\mathcal{K}$ -objects  $A$ ,  $F(id_A) = id_{F(A)}$ ,
- for all  $\mathcal{K}$ -morphisms  $f : A \rightarrow B$  and  $g : B \rightarrow C$ ,  $F(g \circ f) = F(g) \circ F(f)$ .

If  $\mathcal{K} = \mathcal{L}$ , then  $F$  is called an **endofunctor** on  $\mathcal{K}$ .

A **contravariant functor**  $F : \mathcal{K} \rightarrow \mathcal{L}$  is a covariant functor  $F : \mathcal{K}^{op} \rightarrow \mathcal{L}$ .

The (sequential) composition of two functors  $F : \mathcal{K} \rightarrow \mathcal{L}$  and  $G : \mathcal{L} \rightarrow \mathcal{M}$  yields the functor  $\textcolor{red}{GF} : \mathcal{K} \rightarrow \mathcal{M}$ : For all  $A \in \mathcal{K} \cup \text{Mor}_{\mathcal{K}}$ ,  $\textcolor{red}{GF}(A) =_{def} G(F(A))$ .

Exercise 5 Show that  $GF$  is a functor. □

## Examples

Given an object  $B \in \mathcal{L}$ , the **constant functor**  $\textcolor{red}{const}_B : \mathcal{K} \rightarrow \mathcal{L}$  maps each object of  $\mathcal{K}$  to  $B$  and each  $\mathcal{K}$ -morphism to  $id_B \in \mathcal{L}(B, B)$ . One often writes  $B$  instead of  $const_B$ .

The **identity functor**  $\text{Id}_{\mathcal{K}} : \mathcal{K} \rightarrow \mathcal{K}$  maps each  $\mathcal{K}$ -object and each  $\mathcal{K}$ -morphism to itself.

Given a subcategory  $\mathcal{L}$  of  $\mathcal{K}$ , the **forgetful functor**  $\text{U} : \mathcal{L} \rightarrow \mathcal{K}$  maps each  $\mathcal{L}$ -object and each  $\mathcal{L}$ -morphism to itself.

Let  $I$  be a set of indices. The **diagonal functor**  $\Delta_{\mathcal{K}}^I : \mathcal{K} \rightarrow \mathcal{K}^I$  maps each  $\mathcal{K}$ -object  $A$  to the  $I$ -tuple  $(A_i)_{i \in I}$  with  $A_i = A$  for all  $i \in I$  and each  $\mathcal{K}$ -morphism  $f$  to the  $I$ -tuple  $(f_i)_{i \in I}$  with  $f_i = f$  for all  $i \in I$ .

The **product functor**  $\prod_{i \in I} : \text{Set}^I \rightarrow \text{Set}$  and the **coproduct functor**  $\coprod_{i \in I} : \text{Set}^I \rightarrow \text{Set}$  map each tuple  $(A_i)_{i \in I}$  of sets to their product  $\prod_{i \in I} A_i$  or coproduct  $\coprod_{i \in I} A_i$  and each tuple  $(f_i : A_i \rightarrow B_i)_{i \in I}$  of functions to their product  $\prod_{i \in I} f_i$  or coproduct  $\coprod_{i \in I} f_i$ , respectively (see [Preliminaries](#)).

Given sets  $I$  and  $J$  of indices, a functor  $F : \text{Set}^I \rightarrow \text{Set}^J$  is **permutative** if for all  $A \in \text{Set}^I$  and  $j \in J$  there is  $i \in I$  such that  $F(A)_j = A_i$ .

Let  $X$  be a set,  $M$  be a commutative monoid and  $\varphi \subseteq M$ . We have already defined (covariant) endofunctors on  $\text{Set}$  in the [Preliminaries](#), namely:

- the **list functors**  $\_^*$  and  $\_^+$ ,
- the **power or reader functor**  $\\_^X$ , called **stream functor** if  $X = \mathbb{N}$ ,

- the **powerset functor**  $\mathcal{P}$ ,
- the **finite-set functor**  $\mathcal{P}_\omega$ ,
- the **bag functor**  $\mathbb{N}^-$ ,
- the **finite-bag functor**  $\mathbb{N}_\omega^-$ ,
- the **weighted-set functor**  $M_\omega^-$ ,
- the **constrained weighted-set functor**  $M_C^-$ ,
- the **probability (distribution) functor**  $\mathcal{D}_\omega$ .

The **exception functor**  $\underline{\phantom{x}} + X : \text{Set} \rightarrow \text{Set}$  maps a set  $A$  to the set  $A + X$  and a function  $f : A \rightarrow B$  to the function

$$\begin{aligned} f + X : A + X &\rightarrow B + X \\ (a, 1) &\mapsto (f(a), 1) \\ (x, 2) &\mapsto (x, 2) \end{aligned}$$

The **copower or writer functor**  $\underline{\phantom{x}} \times X : \text{Set} \rightarrow \text{Set}$  combines the identity functor with a product functor: It maps a set  $A$  to the set  $A \times X$  and a function  $f : A \rightarrow B$  to the function

$$\begin{aligned} f \times X : A \times X &\rightarrow B \times X \\ (a, x) &\mapsto (f(a), x) \end{aligned}$$

Let  $X$  represent a set of states. The **state functor** (also called **side-effects functor**)

$$(\underline{\phantom{x}} \times X)^X : Set \rightarrow Set$$

sequentially combines a writer functor with a reader functor: It maps a set  $A$  to the set  $(A \times X)^X$  and a function  $f : A \rightarrow B$  to the function

$$\begin{aligned} (f \times X)^X : (A \times X)^X &\rightarrow (B \times X)^X \\ g &\mapsto \lambda(a, x). (f(a), x) \circ g \end{aligned}$$

The **costate functor**

$$(\underline{\phantom{x}}^X \times X : Set \rightarrow Set$$

sequentially combines a reader functor with a writer functor: It maps a set  $A$  to the set  $(A^X) \times X$  and a function  $f : A \rightarrow B$  to the function

$$\begin{aligned} f^X \times X : A^X \times X &\rightarrow B^X \times X \\ (g, x) &\mapsto (f \circ g, x) \end{aligned}$$

If  $X = \mathbb{N}$ , then  $A^X \times X$  represents the set of pairs of a stream  $s$  and a position of  $s$ .

The **labelled-tree functor**  $LT(X) : \text{Set} \rightarrow \text{Set}$  maps a set  $A$  to the set of labelled trees over  $(X, A)$  and a function  $f : A \rightarrow B$  to the function

$$\begin{aligned} LT(X)(f) : ltr(X, A) &\rightarrow ltr(X, B) \\ t &\mapsto f \circ t \end{aligned}$$

The **pointed-tree functor**  $Ptree(X) : \text{Set} \rightarrow \text{Set}$  combines  $LT(X)$  with a writer functor. It maps a set  $A$  to  $ltr(X, A) \times X^*$  and a function  $f : A \rightarrow B$  to

$$\begin{aligned} Ptree(X)(f) : ltr(X, A) \times X^* &\rightarrow ltr(X, B) \times X^* \\ (t, w) &\mapsto (f \circ t, w) \end{aligned}$$

The contravariant (!) **coreader functor**  $X^- : \text{Set} \rightarrow \text{Set}$  maps a set  $A$  to the set  $X^A$  and a function  $f : A \rightarrow B$  to the function

$$\begin{aligned} X^f : X^B &\rightarrow X^A \\ g &\mapsto g \circ f \end{aligned}$$

The **hom-functor**  $\mathcal{K}(\_, \_): \mathcal{K}^{op} \times \mathcal{K} \rightarrow Set$  is defined on morphisms as follows:

For all  $(f: A \rightarrow C, g: B \rightarrow D) = (f, g): (A, B) \rightarrow (C, D) \in Mor(\mathcal{K}^{op} \times \mathcal{K})$ ,

$$\mathcal{K}(f, g) =_{def} \lambda h. g \circ h \circ f: \mathcal{K}(A, B) \rightarrow \mathcal{K}(C, D).$$

Note that  $f \in \mathcal{K}^{op}(A, C)$  implies  $f \in \mathcal{K}(C, A)$  and thus  $g \circ h \circ f \in \mathcal{K}(C, D)$ .

The category **Cat** has categories  $\mathcal{K}$  as objects and functors  $F: \mathcal{K} \rightarrow \mathcal{L}$  as morphisms.

Given two functors  $F, G: \mathcal{K} \rightarrow \mathcal{L}$ , a **natural transformation**  $\tau: F \rightarrow G$  assigns to each object  $A \in \mathcal{K}$  an  $\mathcal{L}$ -morphism  $\tau_A: F(A) \rightarrow G(A)$  such that for all  $\mathcal{K}$ -morphisms  $f: A \rightarrow B$  the following diagram commutes:

$$\begin{array}{ccc}
 F(A) & \xrightarrow{\tau_A} & G(A) \\
 \downarrow F(f) & (2) & \downarrow G(f) \\
 F(B) & \xrightarrow{\tau_B} & G(B)
 \end{array}$$

If for all  $A \in \mathcal{K}$ ,  $\tau_A$  is an isomorphism, then  $\tau: F \rightarrow G$  is a **natural equivalence** and  $F$  and  $G$  are **naturally equivalent**.

## Examples

### 1. The Haskell function

`concat :: [[a]] -> [a]`

is a natural transformation from the composition  $\underline{\phantom{x}}^{**}$  of the list functor with the list functor to the list functor. *concat* is the multiplication of the list monad (see chapter 23).

Exercise 6 Show that  $\tau = concat$  satisfies (2).

### 2. The Haskell function

`uncurry (++) :: ([a], [a]) -> [a]`

is a natural transformation from the composition  $Set \xrightarrow{*} Set \xrightarrow{\Delta_{Set}^2} Set^2 \xrightarrow{\times} Set$  of functors to the functor  $Set \xrightarrow{*} Set$ .

Exercise 7 Show that  $\tau = uncurry(++)$  satisfies (2).

Exercise 8  $\tau : LT(X) \rightarrow \mathcal{P}$  defined by  $\tau_A(t) = \{t(w) \mid w \in def(t)\}$  for all sets  $A$  and  $t \in ltr(X, A)$  satisfies (2).

3. Let  $T : Set \rightarrow Set$  be a functor and  $A$  be a set. The **strength**

$$st^{T,A} : T(-^A) \rightarrow T(-)^A$$

of  $T$  and  $A$  is defined as follows (see [79], p. 380): For all sets  $B$ ,  $g \in T(B^A)$  and  $a \in A$ ,

$$st_B^{T,A}(g)(a) = T(h)(g) \in T(B)$$

where  $h = \lambda f.f(a) : B^A \rightarrow B$ .

$st^{T,A}$  is a natural transformation, i.e., for all  $h : B \rightarrow C$ , (3) commutes:

$$\begin{array}{ccc} T(B^A) & \xrightarrow{st_B^{T,A}} & T(B)^A \\ T(h^A) \downarrow & (3) & \downarrow T(h)^A \\ T(C^A) & \xrightarrow{st_C^{T,A}} & T(C)^A \end{array}$$

*Proof.* At first, we show:

$$(\lambda f.f(a)) \circ h^A = \lambda f.h(f(a)). \quad (4)$$

For all  $a \in A$  and  $g \in B^A$ ,

$$(\lambda f.f(a))(h^A(g)) = (\lambda f.f(a))(h \circ g) = h(g(a)) = (\lambda f.h(f(a)))(g).$$

Hence (3) commutes: For all  $g \in T(B^A)$  and  $a \in A$ ,

$$\begin{aligned}
 (T(h)^A \circ st_B^{T,A}(g))(a) &= (T(h)^A \circ \lambda a.T(\lambda f.f(a))(g))(a) = T(h)^A(T(\lambda f.f(a))(g)) \\
 &= (T(h) \circ T(\lambda f.f(a)))(g) = T(h \circ \lambda f.f(a))(g) = T(\lambda f.h(f(a)))(g), \\
 st_C^{T,A}(T(h^A)(g))(a) &= T(\lambda f.f(a))(T(h^A)(g)) = (T(\lambda f.f(a)) \circ T(h^A))(g) \\
 &= T((\lambda f.f(a)) \circ h^A)(g) \stackrel{(4)}{=} T(\lambda f.h(f(a)))(g)
 \end{aligned}
 \quad \square$$

An object  $A \in \mathcal{K}$  **represents** a covariant or contravariant functor  $F : \mathcal{K} \rightarrow Set$  if  $F$  is naturally equivalent to  $\mathcal{K}(A, \underline{\phantom{x}})$  or  $\mathcal{K}(\underline{\phantom{x}}, A)$ , respectively.

If  $\mathcal{K} = Set$ , then  $\mathcal{K}(A, \underline{\phantom{x}})$  and  $\mathcal{K}(\underline{\phantom{x}}, A)$  agree with the reader functor  $\underline{\phantom{x}}^A$  and the coreader functor  $A-$  (see above).

## Examples

$1$  represents  $Id_{Set}$  (see [135], section 4.2.1).

The monoid  $(\mathbb{N}, \{0, (+)\})$  represents the forgetful functor  $U : Monoid \rightarrow Set$  (see section 19.2 and [135], section 4.2.2).

$\mathbb{N}$  represents the stream functor  $\underline{\phantom{x}}^{\mathbb{N}}$  because for all sets  $A$ ,  $A^{\mathbb{N}}$  is equal to and thus isomorphic to  $Set(\mathbb{N}, A)$ .

## Compositions of natural transformations with functors

- Let  $F, G : \mathcal{K} \rightarrow \mathcal{L}$ ,  $\tau : F \rightarrow G$  and  $H : \mathcal{L} \rightarrow \mathcal{M}$ .

Then  $H\tau : HF \rightarrow HG$  and for all  $A \in \mathcal{K}$ ,  $(H\tau)_A = H(\tau_A) : HF(A) \rightarrow HG(A)$ .

- Let  $F : \mathcal{K} \rightarrow \mathcal{L}$ ,  $G, H : \mathcal{L} \rightarrow \mathcal{M}$  and  $\tau : G \rightarrow H$ .

Then  $\tau F : GF \rightarrow HF$  and for all  $A \in \mathcal{K}$ ,  $(\tau F)_A = \tau_{F(A)} : GF(A) \rightarrow HF(A)$ .

- *Vertical Composition*

Let  $F, G, H : \mathcal{K} \rightarrow \mathcal{L}$ ,  $\tau : F \rightarrow G$  and  $\eta : G \rightarrow H$ .

Then  $\eta\tau : F \rightarrow H$  and for all  $A \in \mathcal{K}$ ,  $(\eta\tau)_A = \eta_A \circ \tau_A : F(A) \rightarrow H(A)$ .

- *Horizontal Composition*

Let  $F, G : \mathcal{K} \rightarrow \mathcal{L}$ ,  $\tau : F \rightarrow G$ ,  $F', G' : \mathcal{L} \rightarrow \mathcal{M}$  and  $\tau' : F' \rightarrow G'$ . Then

$$\tau'\tau : F'F \rightarrow G'G = F'F \xrightarrow{F'\tau} F'G \xrightarrow{\tau'G} G'G = F'F \xrightarrow{\tau'F} G'F \xrightarrow{G'\tau} G'G.$$

Given two categories  $\mathcal{K}$  and  $\mathcal{L}$ , the category  $\text{Fun}(\mathcal{K}, \mathcal{L})$  has all functors  $F : \mathcal{K} \rightarrow \mathcal{L}$  as objects and all natural transformations between such functors and their vertical compositions as morphisms.

## Yoneda Lemma

For all (covariant) functors  $F : \mathcal{K} \rightarrow \text{Set}$  and objects  $A \in \mathcal{K}$ ,

$$\text{Fun}(\mathcal{K}, \text{Set})(\mathcal{K}(A, \_), F) \cong F(A). \quad (4)$$

For all contravariant functors  $F : \mathcal{K} \rightarrow \text{Set}$  and objects  $A \in \mathcal{K}$ ,

$$\text{Fun}(\mathcal{K}, \text{Set})(\mathcal{K}(\_, A), F) \cong F(A). \quad (5)$$

*Proof.* (4): The functions

$$\Phi : \text{Fun}(\mathcal{K}, \text{Set})(\mathcal{K}(A, \_), F) \rightarrow F(A)$$

and

$$\Psi : F(A) \rightarrow \text{Fun}(\mathcal{K}, \text{Set})(\mathcal{K}(A, \_), F)$$

are defined as follows:

For all natural transformations  $\tau : \mathcal{K}(A, \_) \rightarrow F$ ,  $\Phi(\tau) = \tau_A(id_A) \in F(A)$  is well-defined because  $\tau_A$  maps from  $\mathcal{K}(A, A)$  to  $F(A)$ .

For all  $x \in F(A)$ , we define  $\Psi(x) : \mathcal{K}(A, \_) \rightarrow F$  as the natural transformation with

$$\Psi(x)_B(h) = F(h)(x) \in F(B)$$

for all  $B \in \mathcal{K}$  and  $h \in \mathcal{K}(A, B)$ .

$\Phi$  is iso with inverse  $g$ : For all  $x \in F(A)$ ,

$$\Phi(\Psi(x)) = \Psi(x)_A(id_A) = F(id_A)(x) = id_{F(A)}(x) = x,$$

and for all  $\tau : \mathcal{K}(A, \_) \rightarrow F$ ,  $B \in \mathcal{K}$  and  $h \in \mathcal{K}(A, B)$ ,

$$\Psi(\Phi(\tau))_B(h) = F(h)(\Phi(\tau)) = F(h)(\tau_A(id_A)) \stackrel{(2)}{=} \tau_B(\mathcal{K}(A, h)(id_A)) = \tau_B(h \circ id_A) = \tau_B(h).$$

(5): Analogously. □

Functors  $F : \mathcal{K} \rightarrow Set$  play a dominant rôle in categorical modeling. For instance,  $\mathcal{K}$  may model a database schema and  $F$  define an instance of the schema in terms of relations (see, e.g., [160], section 4.5). In fact, “database schema” and “schema instance” are just other words for “signature” and “algebra”, respectively (see chapters 11 and 12).

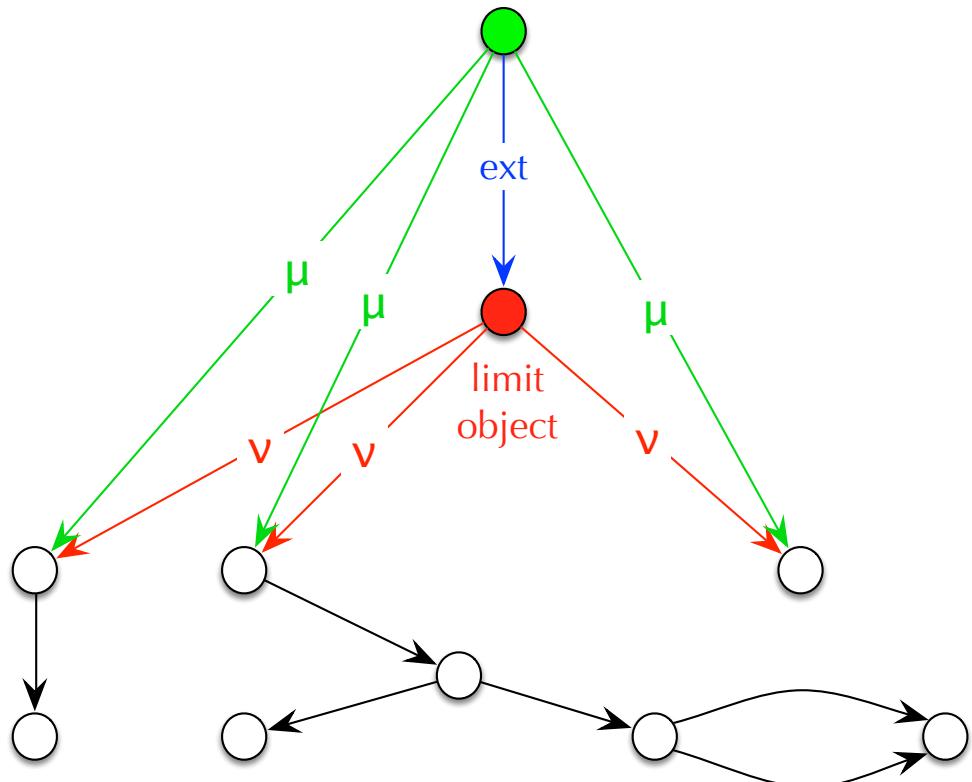
## Limits and colimits

Given two categories  $\mathcal{I}$  and  $\mathcal{K}$ , a  **$\mathcal{K}$ -diagram** is a functor  $\mathcal{D} : \mathcal{I} \rightarrow \mathcal{K}$ , often given as the tuples  $(\mathcal{D}(i))_{i \in \mathcal{I}}$  and  $(\mathcal{D}(f) : \mathcal{D}(i) \rightarrow \mathcal{D}(j))_{f : i \rightarrow j \in \text{Mor}(\mathcal{I})}$ .

The actual objects and morphisms in  $\mathcal{I}$  are irrelevant, only the way in which they are interrelated matters.

One may also view  $\mathcal{D}$  as the node- or edge-labelling function of a labelled graph whose nodes and edges are the objects or morphisms of  $\mathcal{I}$ , respectively.

# Limits



*A diagram, its limit and a further cone*

A tuple  $\mu = (\mu_n : C \rightarrow \mathcal{D}(n))_{n \in \mathcal{I}}$  of  $\mathcal{K}$ -morphisms is a **cone of  $\mathcal{D}$**  if for all  $e \in \mathcal{I}(m, n)$ ,  $\mathcal{D}(e) \circ \mu_m = \mu_n$ .  $C$  is called the **source** of  $\mu$ . A cone is usually identified with its source.

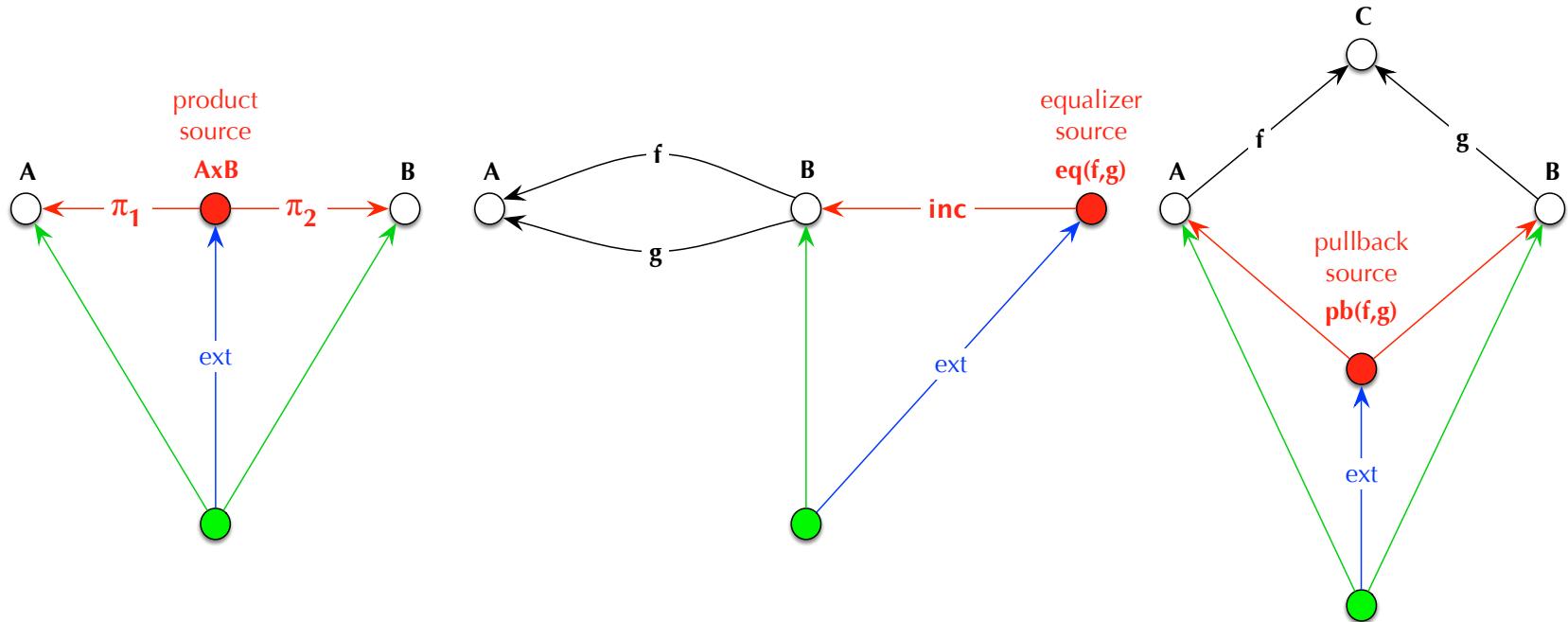
A cone  $\nu$  of  $\mathcal{D}$  with source  $D$  is a **limit of  $\mathcal{D}$**  if for all cones  $\mu = (\mu_n : C \rightarrow \mathcal{D}(n))_{n \in \mathcal{I}}$  of  $\mathcal{D}$  there is a unique  $\mathcal{K}$ -morphism  $\text{ext}: C \rightarrow D$  such that for all  $n \in \mathcal{I}$ ,  $\nu_n \circ \text{ext} = \mu_n$ .

All limits of  $\mathcal{D}$  are isomorphic.

Every object that is isomorphic to a limit of  $\mathcal{D}$  is limit of  $\mathcal{D}$ .

An object is final in  $\mathcal{K}$  if it is the source of a limit of the empty diagram  $\emptyset \rightarrow \mathcal{K}$ .

$\mathcal{K}$  is **complete** if each  $\mathcal{K}$ -diagram has a limit.



Three limits

$$pb(f,g) \cong eq(f \circ \pi_1, g \circ \pi_2).$$

If  $C$  is final in  $\mathcal{K}$ , then  $pb(f,g) = A \times B$ .

The unique morphism  $inc$  from  $eq(f,g)$  is mono.

Let  $A \times B$  be (the source of) a product (object) of  $A$  and  $B$ ,  $F$  be final in  $\mathcal{K}$  and  $I$  be initial in  $\mathcal{K}$ .

Since all products with the same factors are isomorphic,

$$A \times (B \times C) \cong (A \times B) \times C \cong A \times B \times C, A \times B \cong B \times A, A \times F \cong A \text{ and } A \times I \cong I.$$

Let  $\mathcal{K} = \text{Set}$ .

The source of an equalizer of  $(f : B \rightarrow A, g : B \rightarrow A)$  is the set  $S = \{b \in B \mid f(b) = g(b)\}$  together with the inclusion map that sends every element of  $B$  to itself.

The source of a pullback of  $(f : A \rightarrow C, g : B \rightarrow C)$  is the relation  $R = \{(a, b) \in A \times B \mid f(a) = g(b)\}$  together with the projections that send every  $(a, b) \in R$  to  $a$  and  $b$ , respectively.

If  $f$  and  $g$  are inclusion maps, then the pullback object is isomorphic to  $A \cap B$ . Indeed, in this case we obtain:

$$\begin{aligned} R &= \{(a, b) \in A \times B \mid f(a) = g(b)\} = \{(a, b) \in A \times B \mid a = b\} \\ &= \{(a, a) \in A \times B \mid a \in A \cap B\} \end{aligned}$$

and thus  $R \cong A \cap B$ .

Let  $\mathcal{I}$  have no arrows. Then (the source of) a limit  $\nu$  of  $\mathcal{D}$  is called a **product** of  $\mathcal{D}$  in  $\mathcal{K}$ , the components of  $\nu$  are called **projections** and  $ext$  is called a **product extension**.

The above section on **Products** deals with products in  $Set$ . All definitions and results presented there—except for the representation of products by Cartesian ones—also apply to  $\mathcal{K}$  instead of  $Set$ .

In particular,  $\mathcal{K}$  has products iff for all nonempty sets  $I$  and tuples  $(A_i)_{i \in I} \in \mathcal{K}^I$  there are  $P \in \mathcal{K}$ ,  $(d_i : P \rightarrow A_i)_{i \in I} \in Mor(\mathcal{K})^I$  and a function

$$\langle \_ \rangle_{i \in I} : \times_{i \in I} \mathcal{K}(B, A_i) \rightarrow \mathcal{K}(B, P)$$

such that for all  $(f_i : B \rightarrow A_i)_{i \in I} \in Mor(\mathcal{K})^I$ ,  $i \in I$  and  $f : A \rightarrow P \in Mor(\mathcal{K})$ ,

$$\begin{aligned} d_i \circ \langle f_i \rangle_{i \in I} &= f_i, \\ \langle d_i \circ f \rangle_{i \in I} &= f. \end{aligned}$$

A functor  $F : \mathcal{K} \rightarrow \mathcal{L}$  is **product-preserving** if for all products  $\pi = (\pi_i : P \rightarrow A_i)_{i \in I}$  in  $\mathcal{K}$ ,  $F(\pi) =_{def} (F(\pi_i) : F(P) \rightarrow F(A_i))_{i \in I}$  is a product in  $\mathcal{L}$ .

## Subset Theorem (construction of limits in *Set*)

Let  $\mathcal{D} : \mathcal{I} \rightarrow \text{Set}$  be a diagram and

$$R = \{a \in \prod_{n \in \mathcal{I}} \mathcal{D}(n) \mid \forall m, n \in \mathcal{I}, e \in \mathcal{I}(m, n) : \mathcal{D}(e)(\pi_m(a)) = \pi_n(a)\}.$$

The cone

$$(R \xrightarrow{\text{inc}_R} \prod_{n \in \mathcal{I}} \mathcal{D}(n) \xrightarrow{\pi_n} \mathcal{D}(n))_{n \in \mathcal{I}}$$

is a limit of  $\mathcal{D}$ . □

For instance, the Subset Theorem provides the following representations of equalizers and pullbacks, respectively:

Let  $R' = \{(b, a) \in B \times A \mid f(b) = a, g(b) = a\}$ . By the Subset Theorem, the cone

$$(R' \xrightarrow{\text{inc}_{R'}} B \times A \xrightarrow{\pi_1} B, R' \xrightarrow{\text{inc}_{R'}} B \times A \xrightarrow{\pi_2} A)$$

is a limit of  $(f : B \rightarrow A, g : B \rightarrow A)$  and thus  $R'$  isomorphic to the equalizer

$$R = \{b \in B \mid f(b) = g(b)\}$$

of  $(f : B \rightarrow A, g : B \rightarrow A)$  that was constructed above.

Let  $R' = \{(a, b, c) \in A \times B \times C \mid f(a) = c, g(b) = c\}$ . By the Subset Theorem, the cone

$$(R' \xrightarrow{\text{inc}_{R'}} A \times B \times C \xrightarrow{\pi_1} C, R' \xrightarrow{\text{inc}_{R'}} A \times B \times C \xrightarrow{\pi_2} A, R' \xrightarrow{\text{inc}_{R'}} A \times B \times C \xrightarrow{\pi_3} B)$$

is a limit of  $(f : A \rightarrow C, g : B \rightarrow C)$  and thus  $R'$  is isomorphic to the pullback

$$R = \{(a, b) \in A \times B \mid f(a) = g(b)\}$$

of  $(f : A \rightarrow C, g : B \rightarrow C)$  that was constructed above.

**Limit Theorem** (generalizes the Subset Theorem to complete categories)

Let  $\mathcal{K}$  be a complete category,  $\mathcal{D} : \mathcal{I} \rightarrow \mathcal{K}$  be a diagram,

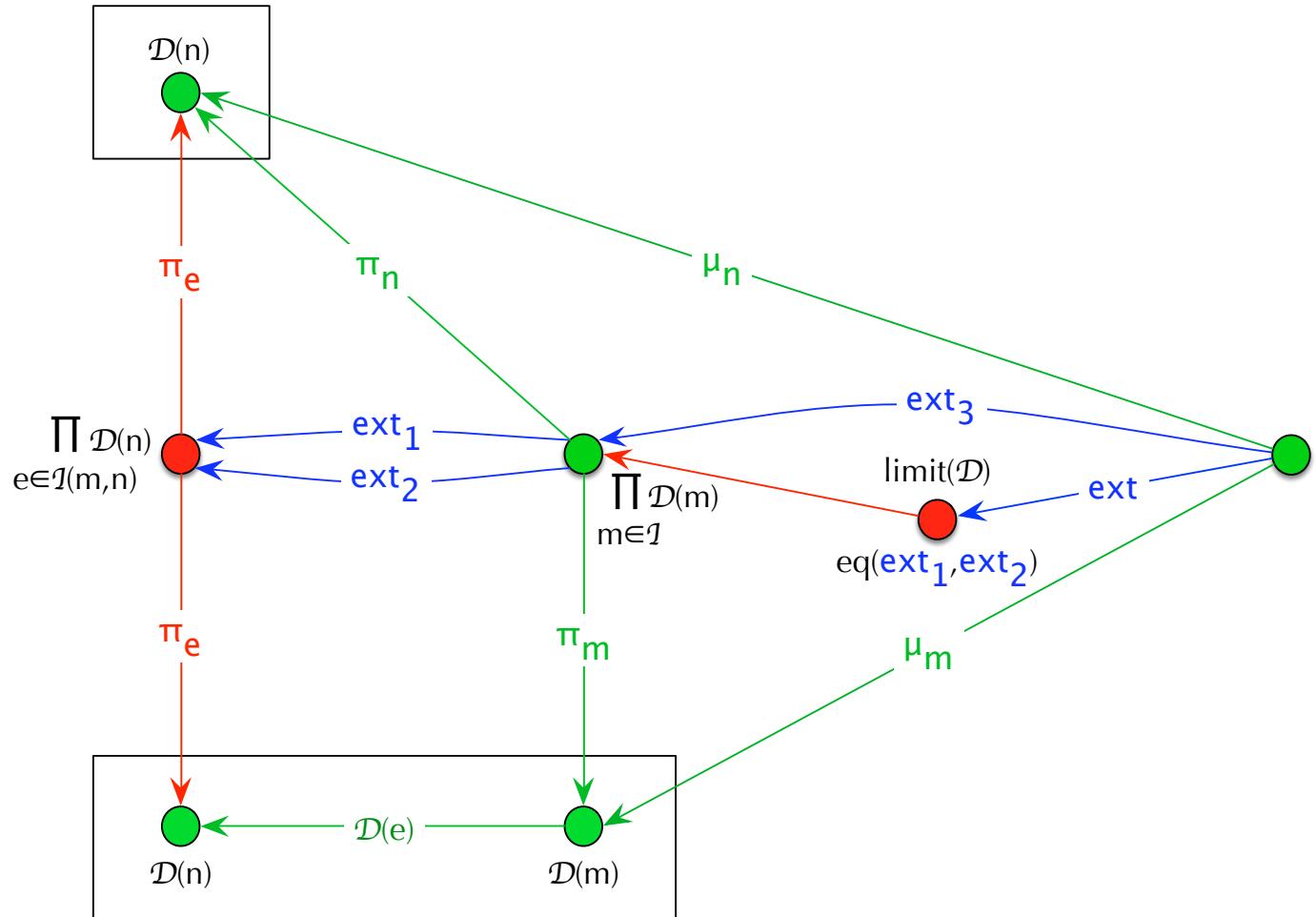
- $(\prod_{m \in \mathcal{I}} \mathcal{D}(m) \xrightarrow{\pi_m} \mathcal{D}(m))_{m \in \mathcal{I}}$  be a product of  $\{\mathcal{D}(m) \mid m \in \mathcal{I}\}$ ,
- $(\prod_{e \in \mathcal{I}(m,n)} \mathcal{D}(n) \xrightarrow{\pi_e} \mathcal{D}(n))_{e \in \mathcal{I}(m,n)}$  be a product of  $\{\mathcal{D}(e) \mid e \in \mathcal{I}(m,n)\}$ ,
- $\prod_{m \in \mathcal{I}} \xrightarrow{\text{ext}_1} \prod_{e \in \mathcal{I}(m,n)} \mathcal{D}(n)$  be the product extension of  

$$\left( \prod_{m \in \mathcal{I}} \mathcal{D}(m) \xrightarrow{\pi_n} \mathcal{D}(n) \right)_{e \in \mathcal{I}(m,n)},$$
- $\prod_{m \in \mathcal{I}} \xrightarrow{\text{ext}_2} \prod_{e \in \mathcal{I}(m,n)} \mathcal{D}(n)$  be the product extension of  

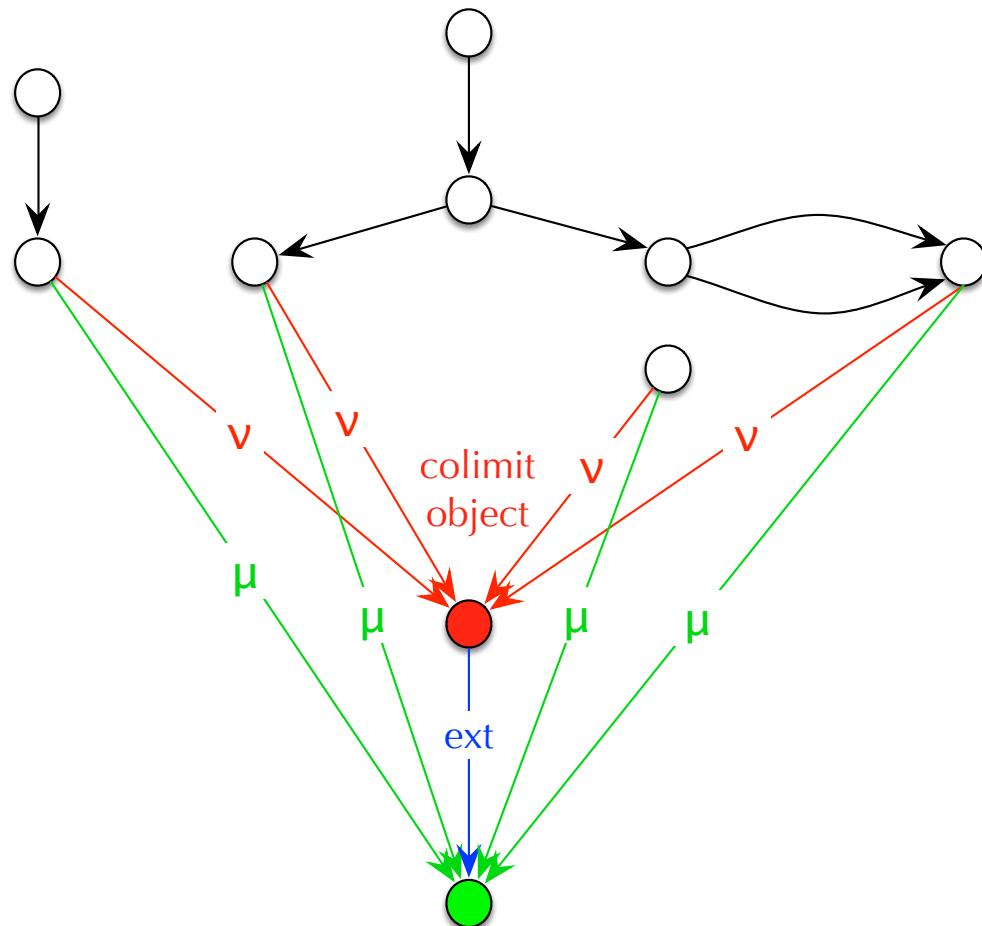
$$\left( \prod_{m \in \mathcal{I}} \mathcal{D}(m) \xrightarrow{\mathcal{D}(e) \circ \pi_m} \mathcal{D}(n) \right)_{e \in \mathcal{I}(m,n)}.$$

The equalizer of  $\{\text{ext}_1, \text{ext}_2\}$  is a limit of  $\mathcal{D}$ .

*Proof.* See the proof of [16], Theorem 2.4.17, or [133], Theorem 1.9.7. □



# Colimits



A diagram, its *colimit* and a further *cone*

Let  $\mathcal{D} : \mathcal{I} \rightarrow \mathcal{K}$  be a  $\mathcal{K}$ -diagram.

A tuple  $\mu = (\mu_n : \mathcal{D}(n) \rightarrow C)_{n \in \mathcal{I}}$  of  $\mathcal{K}$ -morphisms is a **cocone of  $\mathcal{D}$**  if for all  $e \in \mathcal{I}(m, n)$ ,  $\mu_m = \mu_n \circ \mathcal{D}(e)$ .  $C$  is called the **target** of  $\mu$ . A cocone is usually identified with its target.

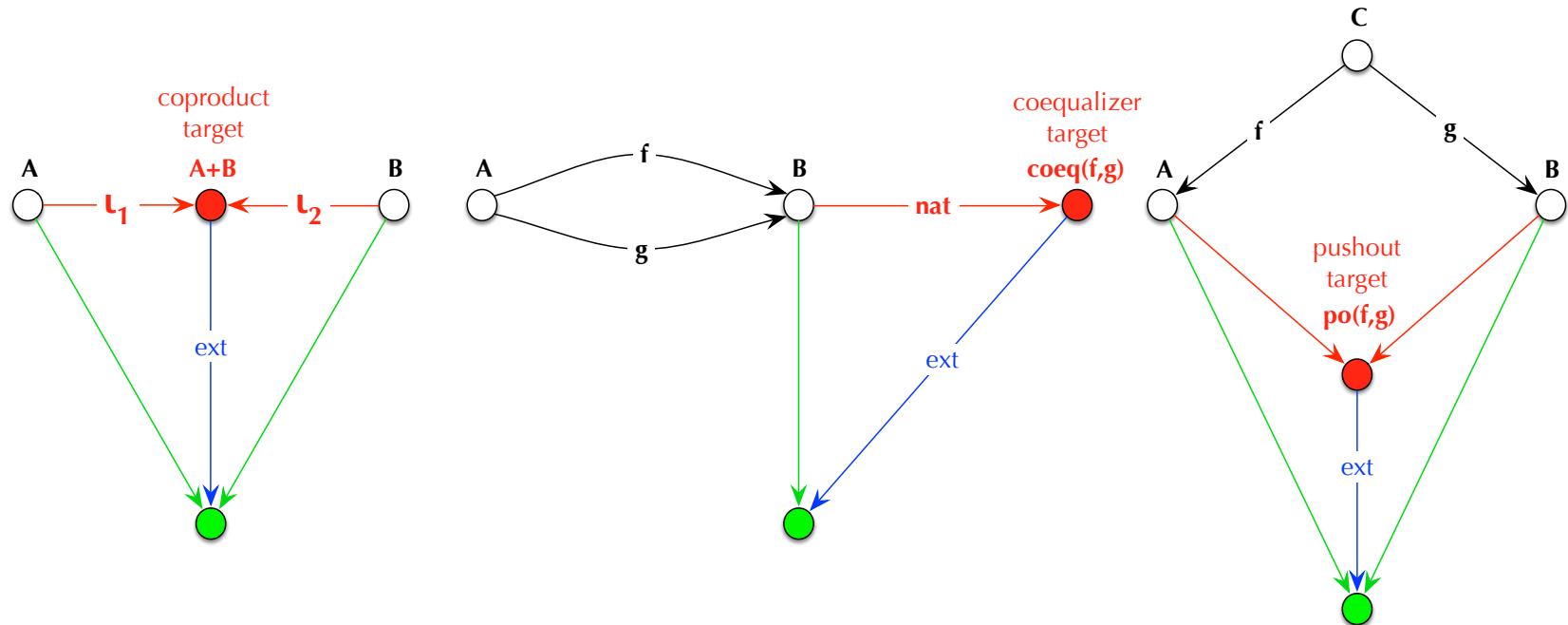
A cocone  $\nu$  of  $\mathcal{D}$  with target  $D$  is a **colimit of  $\mathcal{D}$**  if for all cocones  $\mu = (\mu_n : \mathcal{D}(n) \rightarrow C)_{n \in \mathcal{I}}$  of  $\mathcal{D}$  there is a unique  $\mathcal{K}$ -morphism  $\text{ext} : D \rightarrow C$  such that for all  $n \in \mathcal{I}$ ,  $\text{ext} \circ \nu_n = \mu_n$ .

All colimits of  $\mathcal{D}$  are isomorphic.

Every object that is isomorphic to a colimit of  $\mathcal{D}$  is colimit of  $\mathcal{D}$ .

An object is initial in  $\mathcal{K}$  if it is the target of a colimit of the empty diagram  $\emptyset \rightarrow \mathcal{K}$ .

$\mathcal{K}$  is **cocomplete** if each  $\mathcal{K}$ -diagram has a colimit.



*Three colimits*

$$po(f, g) \cong coeq(\iota_1 \circ f, \iota_2 \circ g).$$

If  $C$  is initial in  $\mathcal{K}$ , then  $po(f, g) \cong A + B$ .

The unique morphism  $nat$  to  $coeq(f, g)$  is epi.

Let  $A + B$  be (the target of) a coproduct of  $A$  and  $B$  and  $I$  be initial in  $\mathcal{K}$ .

Since all coproducts with the same summands are isomorphic,

$$A + (B + C) \cong (A + B) + C \cong A + B + C, \quad A + B \cong B + A \quad \text{and} \quad I + A \cong A.$$

Let  $\mathcal{K} = \text{Set}$ .

The target of a coequalizer of  $(f : A \rightarrow B, g : A \rightarrow B)$  is the quotient of  $B$  by the equivalence closure of  $R = \{(f(a), g(a)) \mid a \in A\}$  together with the natural map that sends every element of  $B$  to its equivalence class w.r.t.  $R^{eq}$ .

The target of a pushout of  $(f : C \rightarrow A, g : C \rightarrow B)$  is the quotient of  $A + B$  by the equivalence closure of  $R = \{(\iota_1(f(c)), \iota_2(g(c))) \mid c \in C\}$  together with the natural maps that send every element of  $A$  or  $B$  to its equivalence class w.r.t.  $R^{eq}$ .

If  $f$  and  $g$  are inclusion maps, then the pushout object is isomorphic to  $A + B$ . Indeed, in this case we obtain:

$$R = \{(\iota_1(f(c)), \iota_2(g(c))) \mid c \in C\} = \{(\iota_1(c), \iota_2(c)) \mid c \in C\}$$

and thus  $(A + B)/R^{eq} \cong A + B$ .

Let  $\mathcal{I}$  have no arrows. Then (the target of) a colimit  $\nu$  of  $\mathcal{D}$  is called a **coproduct** or **sum** of  $\mathcal{D}$  in  $\mathcal{K}$ , the components of  $\nu$  are called **injections** and  $\text{ext}$  is called a **sum extension**.

The above section on **Sums** deals with sums in  $\text{Set}$ . All definitions and results presented there—except for the representation of sums by disjoint unions—also apply to  $\mathcal{K}$  instead of  $\text{Set}$ .

In particular,  $\mathcal{K}$  has sums iff for all nonempty sets  $I$  and tuples  $(A_i)_{i \in I} \in \mathcal{K}^I$  there are  $S \in \mathcal{K}$ ,  $(c_i : A_i \rightarrow S)_{i \in I} \in \text{Mor}(\mathcal{K})^I$  and a function

$$[\_]_{i \in I} : \bigtimes_{i \in I} \mathcal{K}(A_i, B) \rightarrow \mathcal{K}(S, B)$$

such that for all  $(f_i : A_i \rightarrow B)_{i \in I} \in \text{Mor}(\mathcal{K})^I$ ,  $i \in I$  and  $f : S \rightarrow B \in \text{Mor}(\mathcal{K})$ ,

$$\begin{aligned} [f_i]_{i \in I} \circ c_i &= f_i, \\ [f \circ c_i]_{i \in I} &= f. \end{aligned}$$

A functor  $F : \mathcal{K} \rightarrow \mathcal{L}$  is **sum-preserving** if for all sums  $\iota = (\iota_i : A_i \rightarrow S)_{i \in I}$  in  $\mathcal{K}$ ,  $F(\iota) =_{\text{def}} (F(\iota_i) : F(A_i) \rightarrow F(S))_{i \in I}$  is a sum in  $\mathcal{L}$ .

## Quotient Theorem (construction of colimits in *Set*)

Let  $\mathcal{D} : \mathcal{I} \rightarrow \text{Set}$  be a diagram and  $\sim$  be the equivalence closure of

$$R = \{(\iota_m(a), \iota_n(\mathcal{D}(e)(a))) \in (\coprod_{n \in \mathcal{I}} \mathcal{D}(n))^2 \mid a \in \mathcal{D}(m), e \in \mathcal{I}(m, n), m, n \in \mathcal{I}\}.$$

The cocone

$$(\mathcal{D}(n) \xrightarrow{\iota_n} \coprod_{n \in \mathcal{I}} \mathcal{D}(n) \xrightarrow{\text{nat}_\sim} (\coprod_{n \in \mathcal{I}} \mathcal{D}(n))/\sim)_{n \in \mathcal{I}}$$

is a colimit of  $\mathcal{D}$ . □

For instance, the Quotient Theorem provides the following representations of coequalizers and pushouts, respectively:

Let  $R' = \{(\iota_1(a), \iota_2(f(a))) \mid a \in A\} \cup \{(\iota_1(a), \iota_2(g(a))) \mid a \in A\}$ ,  $\sim = R'^{\text{eq}}$  and  $S = (A + B)/\sim$ . By the Quotient Theorem, the cocone

$$(A \xrightarrow{\iota_1} A + B \xrightarrow{\text{nat}_\sim} S, B \xrightarrow{\iota_2} A + B \xrightarrow{\text{nat}_\sim} S)$$

is a colimit of  $(f : A \rightarrow B, g : A \rightarrow B)$  and thus  $S$  is isomorphic to the coequalizer  $B/R^{\text{eq}}$  of  $(f : A \rightarrow B, g : A \rightarrow B)$  with  $R = \{(f(a), g(a)) \mid a \in A\}$  that was constructed above.

Let  $R' = \{(\iota_3(c), \iota_1(f(c))) \mid c \in C\} \cup \{(\iota_3(c), \iota_2(g(c))) \mid c \in C\}$ ,  $\sim = R'^{eq}$  and  $S = (A + B + C)/\sim$ . By the Quotient Theorem, the cocone

$$(A \xrightarrow{\iota_1} A + B + C \xrightarrow{nat} S, B \xrightarrow{\iota_2} A + B + C \xrightarrow{nat} S, C \xrightarrow{\iota_3} A + B + C \xrightarrow{nat} S)$$

is a colimit of  $(f : C \rightarrow A, g : C \rightarrow B)$  and thus  $S$  is isomorphic to the pushout  $(A + B)/R^{eq}$  of  $(f : C \rightarrow A, g : C \rightarrow B)$  with  $R = \{(\iota_1(f(c)), \iota_2(g(c))) \mid c \in C\}$  that was constructed above.

**Colimit Theorem** (generalizes the Quotient Theorem to cocomplete categories)

Let  $\mathcal{K}$  be a cocomplete category,  $\mathcal{D} : \mathcal{I} \rightarrow \mathcal{K}$  be a diagram,

- $(\mathcal{D}(n) \xrightarrow{\iota_n} \coprod_{n \in \mathcal{I}} \mathcal{D}(n))_{n \in \mathcal{I}}$  be a coproduct of  $\{\mathcal{D}(n) \mid n \in \mathcal{I}\}$ ,
- $(\mathcal{D}(m) \xrightarrow{\iota_e} \coprod_{e \in \mathcal{I}(m,n)} \mathcal{D}(m))_{e \in \mathcal{I}(m,n)}$  be a coproduct of  $\{\mathcal{D}(e) \mid e \in \mathcal{I}(m,n)\}$ ,
- $\coprod_{e \in \mathcal{I}(m,n)} \mathcal{D}(m) \xrightarrow{\text{ext}_1} \coprod_{n \in \mathcal{I}} \mathcal{D}(n)$  be the sum extension of

$$(\mathcal{D}(m) \xrightarrow{\iota_m} \coprod_{n \in \mathcal{I}} \mathcal{D}(n))_{e \in \mathcal{I}(m,n)},$$

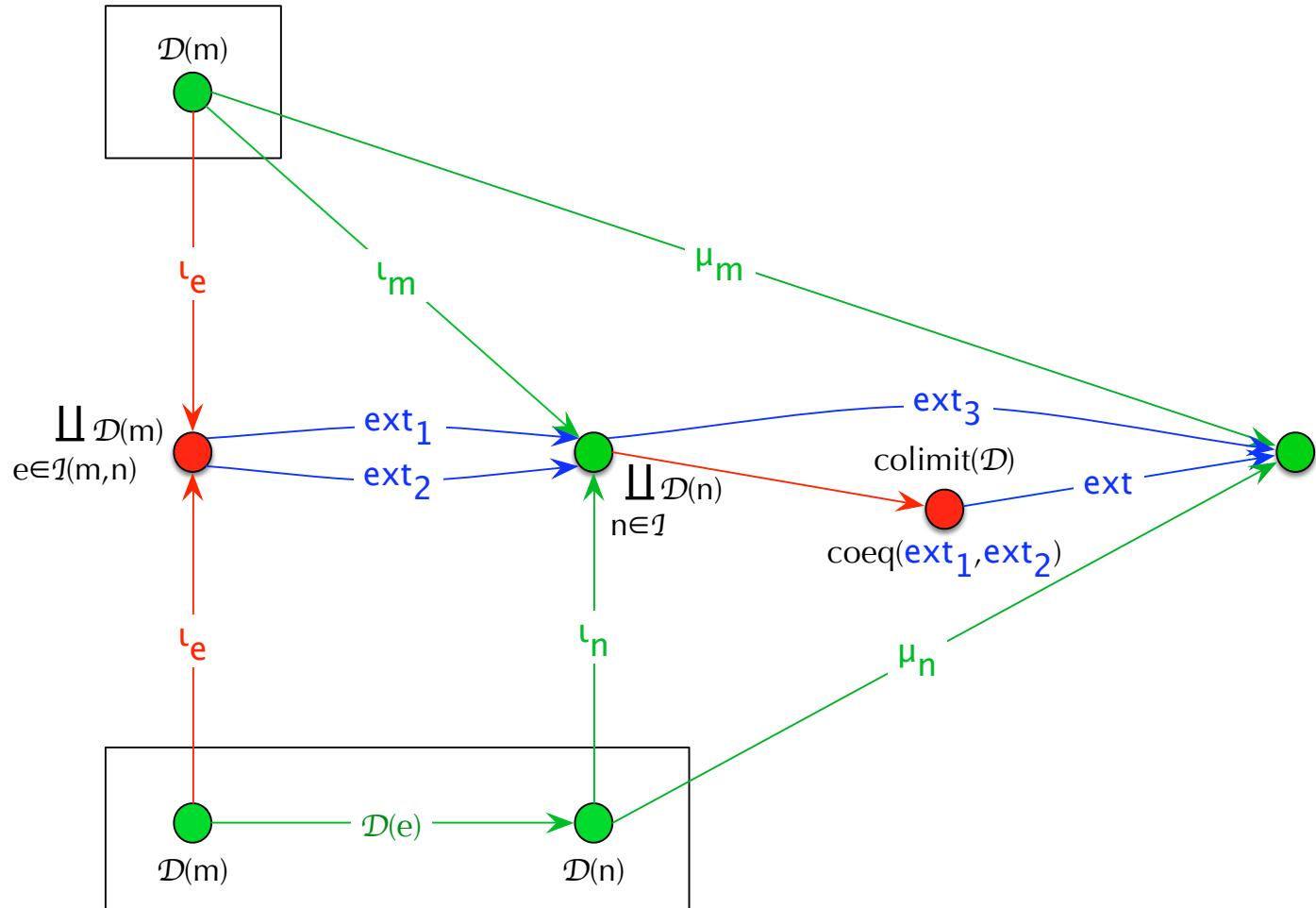
- $\coprod_{e \in \mathcal{I}(m,n)} \mathcal{D}(m) \xrightarrow{\text{ext}_2} \coprod_{n \in \mathcal{I}} \mathcal{D}(n)$  be the sum extension of

$$(\mathcal{D}(m) \xrightarrow{\iota_n \circ \mathcal{D}(e)} \coprod_{n \in \mathcal{I}} \mathcal{D}(n))_{e \in \mathcal{I}(m,n)}.$$

The coequalizer of  $\{\text{ext}_1, \text{ext}_2\}$  is a colimit of  $\mathcal{D}$ .

*Proof.*

Since  $\mathcal{K}^{op}$  is complete, the theorem follows from the Limit Theorem by dualization.  $\square$



## Sorted sets and types

Let  $S$  be a nonempty set. An  **$S$ -sorted** or  **$S$ -indexed set** is a tuple  $A = (A_s)_{s \in S}$  of sets. Sometimes we write  $A$  for the union of  $A_s$  over all  $s \in S$ .  $a \in A_s$  is often written as  $a : s \in A$ .

Let  $A = (A_s)_{s \in S}$  and  $B = (B_s)_{s \in S}$  be  $S$ -sorted sets.

An  **$S$ -sorted function**  $f : A \rightarrow B$  is a tuple  $(f_s : A_s \rightarrow B_s)_{s \in S}$  of functions.

$B^A$  denotes the set of  $S$ -sorted functions from  $A$  to  $B$ .

An  $S$ -sorted function  $f$  is epi/mono/iso iff for all  $s \in S$ ,  $f_s$  is surjective/injective/bijective.

A **multivalued  $S$ -sorted function**  $f : A \rightarrow B$  is a tuple  $(f_s : A_s \rightarrow B_s)_{s \in S}$  of multi-valued functions (see chapter 4).

$\text{Set}^S$  denotes the (product) category of  $S$ -sorted sets as objects and  $S$ -sorted functions as morphisms. Composition and identities are defined componentwise.

$\text{Set}_{\mathcal{P}}^S$  denotes the (product) category of  $S$ -sorted sets as objects and multivalued  $S$ -sorted functions as morphisms. Composition and identities are defined componentwise.

The  $S$ -sorted set  $A$  with  $A_s = \emptyset$  for all  $s \in S$  is initial in  $\text{Set}^S$  and initial and final in  $\text{Set}_{\mathcal{P}}^S$ .

Every  $S$ -sorted set  $A$  with  $|A_s| = 1$  for all  $s \in S$  is final in  $\text{Set}^S$ .

For all ordinal numbers  $\lambda$ ,  $\text{Set}^S$  is  $\lambda$ -complete and  $\lambda$ -cocomplete.

Given  $S$ -sorted sets  $A_1, \dots, A_n$ ,

$$A = \color{red}{A_1 \times \dots \times A_n} =_{\text{def}} (A_{1,s} \times \dots \times A_{n,s})_{s \in S}$$

is called an  **$S$ -sorted product**.

An  **$S$ -sorted  $n$ -ary relation on  $A$**  is an  $S$ -sorted set  $R = (R_s)_{s \in S}$  such that for all  $s \in S$ ,

$$R_s \subseteq A_{1,s} \times \dots \times A_{n,s}.$$

If  $n = 1$ , then  $R$  is also called an  **$S$ -sorted subset of  $A$** . Besides subsets, *binary* relations ( $n = 2$ ) are the most important relations in algebra (see [Products, sums, et al.](#)).

Let  $\mathcal{I}$  be an **alphabet**, i.e., a set of given constants including () and the natural numbers.  $\mathcal{I}$  provides the **indices** of sum and product types, subsets of  $\mathcal{I}$  denote **primitive types** [176, 177], which, in some other approaches, would be called *parameter types* [27].

The sets  $\mathcal{T}_p(S, \mathcal{I})$ ,  $\mathcal{T}_q(S, \mathcal{I})$  and  $\mathcal{T}(S, \mathcal{I})$  of **polynomial**, **quantitative** resp. **exponential types over**  $(S, \mathcal{I})$  are inductively defined as follows: Let  $k \in \{p, e\}$ .

- $S \cup \mathcal{I} \subseteq \mathcal{T}_p(S, \mathcal{I})$ , (sorts and indices)
- $\mathcal{T}_p(S, \mathcal{I}) \subseteq \mathcal{T}_q(S, \mathcal{I})$ , (polynomial types are quantitative)
- $\mathcal{T}_q(S, \mathcal{I}) \subseteq \mathcal{T}(S, \mathcal{I})$ , (quantitative types are exponential)
- for all  $I \subseteq \mathcal{I}$  and  $(e_i)_{i \in I} \in \mathcal{T}_k(S, \mathcal{I})^I$ ,  $\coprod_{i \in I} e_i \in \mathcal{T}_k(S, \mathcal{I})$ , (sum types)
- for all  $I \subseteq \mathcal{I}$  and  $(e_i)_{i \in I} \in \mathcal{T}_k(S, \mathcal{I})^I$ ,  $\prod_{i \in I} e_i \in \mathcal{T}_k(S, \mathcal{I})$ , (product types)
- for all  $e \in \mathcal{T}_p(S, \mathcal{I})$ , commutative monoids  $(M, +, 0)$  and  $C \subseteq M$ ,  
 $(M \times e)_C^* \in \mathcal{T}_p(S, \mathcal{I})$  and  $M_C^e \in \mathcal{T}_q(S, \mathcal{I})$ , (weighted types)
- for all  $e, e' \in \mathcal{T}(S, \mathcal{I})$ ,  $e \rightarrow e' \in \mathcal{T}(S, \mathcal{I})$ . (function types)
- for all  $e \in \mathcal{T}(S, \mathcal{I})$ ,  $\mathcal{P}(e) \in \mathcal{T}(S, \mathcal{I})$ . (powerset types)

For all  $i \in I$ ,  $e_i$  is called a **summand** of the sum type  $\coprod_{i \in I} e_i$  and a **factor** of the product type  $\prod_{i \in I} e_i$ .  $() \in \mathcal{I}$  is called the **unit type**.

Final models of weighted types of the form  $M_C^e$  are quotients of polynomial weighted types, i.e., types of the form  $(M \times e)_C^*$  (see chapter 13).

A function type  $e \rightarrow e'$  is called an **arrow type** if  $e, e' \in \mathcal{T}_q(S, \mathcal{I})$  (see chapter 8).

Given  $e \in \mathcal{T}_p(S, \mathcal{I})$  and  $(e_s)_{s \in S} \in \mathcal{T}(S, \mathcal{I})^S$ ,  $e[e_s/s \mid s \in S]$  denotes the type obtained from  $e$  by replacing all occurrences of  $s \in S$  with  $e_s$ .

On the one hand, types over  $(S, \mathcal{I})$  generalize classical *regular expressions* over  $\mathcal{I}$  by admitting sums and products with arbitrary index sets  $I \subseteq \mathcal{I}$ .

On the other hand, types over  $(S, \mathcal{I})$  may be regarded as *dependent types* that take the indices of sum and product types from  $\mathcal{I}$ . The notations, however, are different: A sum type  $\coprod_{i \in I} e_i$  of  $\mathcal{T}(S, \mathcal{I})$  corresponds to a *dependent pair type* and is written as  $\sum i : I.e_i$ , while  $\prod_{i \in I} e_i$  resembles a *dependent function type* and is written as  $\prod i : I.e_i$  (see, e.g., [13]). “Pair” and “function” probably refer to the usual representation of elements of sums and products as disjoint unions and Cartesian products, respectively. Since type theorists would not regard  $I$  as a set, the expression  $i : I$  does not necessarily denote set membership as  $i \in I$  does. Hence the differences are not only notational.

## Derived types

For all  $I \subseteq \mathcal{I}$ ,  $n > 0$ ,  $e, e_1, \dots, e_n \in \mathcal{T}(S, \mathcal{I})$ , commutative monoids  $(M, +, 0)$  and  $e \in \mathcal{T}_p(S, \mathcal{I})$ ,

$I$	$= \coprod_{i \in I} i,$	(primitive types)
$e^I$	$= \prod_{i \in I} e,$	(power types)
$e_1 + \dots + e_n$	$= \coprod_{i \in [n]} e_i,$	(finite sums)
$e_1 \times \dots \times e_n$	$= \prod_{i \in [n]} e_i,$	(finite products)
$e^0$	$= () ,$	
$e^n$	$= e^{[n]},$	
$e^+$	$= \coprod_{n > 0} e^n,$	(nonempty finite lists)
$e^*$	$= \coprod_{n \in \mathbb{N}} e^n,$	(finite lists)
$e^\infty$	$= e^* + e^{\mathbb{N}},$	(finite or infinite lists)
$\mathcal{P}_\omega(e)$	$= 2_2^e,$	(finite sets)
$\mathcal{D}(e)$	$= (\mathbb{R}_{\geq 0})_{\{1\}}^e.$	(probabilistic types)

## Type models

A  $\mathcal{T}_p(S, \mathcal{I})$ -sorted set  $A$  is a **polynomial model over**  $(S, \mathcal{I})$  if

- for all  $i \in \mathcal{I}$ ,  $A_i \cong \{i\}$ ,
- for all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $A_e \cong \biguplus_{i \in I} A_{e_i}$ ,
- for all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $A_e \cong \bigtimes_{i \in I} A_{e_i}$ ,
- for all weighted types  $(M \times e)_C^* \in \mathcal{T}_p(S, \mathcal{I})$ ,

$$A_{(M \times e)_C^*} \cong \{((m_1, a_1), \dots, (m_n, a_m)) \in (M \times A_e)^+ \mid \sum_{i=1}^n m_i \in C\}.$$

A  $\mathcal{T}_q(S, \mathcal{I})$ -sorted set  $A$  is a **quantitative model over**  $(S, \mathcal{I})$  if

- $A$  is a polynomial model over  $(S, \mathcal{I})$ ,
- for all weighted types  $M_C^e \in \mathcal{T}_q(S, \mathcal{I})$ ,

$$A_{M_C^e} \cong M_{\omega, C}^{A_e} = \{f : A_e \rightarrow_{\omega} M \mid \sum_{a \in A_e} f(a) \in C\}$$

(see Preliminaries).

We often use notations for elements of  $A_e$  as if  $A_e$  would have the “standard” representation given by the set on the right-hand side of the respective isomorphism, e.g., as if  $A_{eI}$  were equal to  $(A_e)^I$  or  $A_{e^*}$  were equal to  $A_e^*$ .

Quantitative models over  $(S, \mathcal{I})$  provide the objects of the subcategory  $\textcolor{red}{Mod}(S, \mathcal{I})$  of  $\textcolor{violet}{Set}^{\mathcal{T}_q(S, \mathcal{I})}$  whose morphisms are the  $\mathcal{T}_q(S, \mathcal{I})$ -sorted functions  $\textcolor{blue}{h} : A \rightarrow \textcolor{blue}{B}$  that satisfy the following conditions:

Let  $I \subseteq \mathcal{I}$ .

- For all  $i \in \mathcal{I}$ ,  $\textcolor{blue}{h}_i = \textcolor{blue}{id}_{\{i\}}$ .  
• For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_q(S, \mathcal{I})$ ,  $\textcolor{blue}{h}_e = \coprod_{i \in I} h_{e_i}$ .  
• For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_q(S, \mathcal{I})$ ,  $\textcolor{blue}{h}_e = \prod_{i \in I} h_{e_i}$ .  
• For all weighted types  $e = (M \times e')_C^* \in \mathcal{T}_p(S, \mathcal{I})$ ,  $\textcolor{blue}{h}_e = h_{(M \times e')^*}|_{A_e}$ .  
• For all weighted types  $e = M_C^{e'} \in \mathcal{T}_q(S, \mathcal{I})$ ,  $\textcolor{blue}{h}_e = M_C^{h_{e'}}$ .  
(1)  
(2)

These equations yield an inductively defined extension of every  $S$ -sorted function to a  $\mathcal{T}_q(S, \mathcal{I})$ -sorted function and thus, for each  $e \in \mathcal{T}_q(S, \mathcal{I})$ , a functor

$$F_e : \text{Mod}(S, \mathcal{I}) \rightarrow \text{Set}$$

that maps every model  $A$  of  $\mathcal{T}_q(S, \mathcal{I})$  to  $A_e$  and every  $\text{Mod}(S, \mathcal{I})$ -morphism  $h$  to  $h_e$ .

For instance,  $h_{s^*} = \coprod_{n \in \mathbb{N}} \prod_{i=1}^n h_s$ , i.e.,  $h_{s^*}$  agrees with  $h_s^* = \text{map}(f_s)$  (see Preliminaries).

The functor property requires that for all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $h_e$  is a function from  $A_e$  to  $B_e$ . This is trivial in all cases except (1) and (2). For (2), see Preliminaries. For (1), a proof reads as follows:

Let  $((m_1, a_1), \dots, (m_n, a_m)) \in A_{(M \times e')_C^*}$ . Then  $\sum_{i=1}^n m_i \in C$ . Hence

$$\begin{aligned} h_{(M \times e')_C^*}((m_1, a_1), \dots, (m_n, a_m)) &= h_{(M \times e')^*}((m_1, a_1), \dots, (m_n, a_m)) \\ &= ((m_1, h_s(a_1)), \dots, (m_n, h_s(a_m))) \in B_{(M \times e')_C^*}. \end{aligned}$$

Two types  $e, e'$  over  $(S, \mathcal{I})$  are **equivalent** if  $F_e$  and  $F_{e'}$  are naturally equivalent.

$e'$  is called a **quotient** of  $e$  if there is a surjective natural transformation from  $F_{e'}$  to  $F_e$ .

For instance, for all  $I \subseteq \mathcal{I}$  and  $e, e_i \in \mathcal{T}_q(S, \mathcal{I})$ ,  $i \in I$ ,

$$e \rightarrow 2^{\coprod_{i \in I} e_i} \text{ and } \prod_{i \in I} (e \rightarrow 2^{e_i}) \text{ are equivalent.} \quad (3)$$

Indeed, the function

$$h : (A \rightarrow \mathcal{P}_\omega(\coprod_{i \in I} A_i)) \rightarrow \prod_{i \in I} (A \rightarrow \mathcal{P}_\omega(A_i))$$

with  $\pi_i(h(f))(a) = \{b \mid (b, i) \in f(a)\}$  for all  $i \in I$ ,  $f : A \rightarrow \mathcal{P}_\omega(\coprod_{i \in I} A_i)$  and  $a \in A$ .  $h$  has an inverse that is defined as follows: For all  $g = (g_i : A \rightarrow \mathcal{P}_\omega(A_i))_{i \in I}$  and  $a \in A$ ,

$$h^{-1}(g)(a) = \{(b, i) \mid i \in I, b \in g_i(a)\}.$$

However, if  $\mathcal{P}_\omega$  is replaced by the list functor, the counterpart

$$h' : (A \rightarrow (\coprod_{i \in I} A_i)^*) \rightarrow \prod_{i \in I} (A \rightarrow A_i^*),$$

of  $h$ , defined by:

$$\pi_i(h'(f))(a) = \text{map}(\pi_1)(\text{filter}(\lambda(b, j).i = j)(f(a)))$$

for all  $i \in I$ ,  $f \in A \rightarrow (\coprod_{i \in I} A_i)^*$  and  $a \in A$  is surjective, but not injective, i.e.,

$$\prod_{i \in I} (e \rightarrow e_i^*) \text{ is only a quotient of } e \rightarrow (\coprod_{i \in I} e_i)^*. \quad (4)$$

A *functional* type  $e \rightarrow e'$  provides only an *object* mapping  $F_{e \rightarrow e'} : Mod(S, \mathcal{I}) \rightarrow Set$ :

- For all  $A \in Mod(S, \mathcal{I})$ ,  $F_{e \rightarrow e'}(A) = A_e \rightarrow A_{e'}$ .

## Lifting of sorted relations

Let  $A_1, \dots, A_n$  be models of  $\mathcal{T}_q(S, \mathcal{I})$ . An  $S$ -sorted  $n$ -ary relation  $R$  on the  $S$ -sorted product  $\mathbf{A} = A_1 \times \dots \times A_n$  (see above) is lifted to a  $\mathcal{T}_q(S, \mathcal{I})$ -sorted  $n$ -ary relation as follows:

- For all  $i \in \mathcal{I}$ ,  $R_i = \Delta_{\{i\}}^{[n]} = \{(i, \dots, i)\}$ ,
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_q(S, \mathcal{I})$ ,

$$R_e = \{(\iota_i(a_j))_{j=1}^n \in \bigtimes_{j=1}^n A_{j,e} \mid (a_j)_{j=1}^n \in R_{e_i}, i \in I\}.$$

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_q(S, \mathcal{I})$ ,

$$R_e = \{(a_k)_{j=1}^n \in \bigtimes_{j=1}^n A_{j,e} \mid \forall i \in I : (\pi_i(a_j))_{j=1}^n \in R_{e_i}\}.$$

- For all weighted types  $e = (M \times e')_C^*$ ,

$$R_e = \{((m_{ij}, a_{ij})_{i=1}^k)_{j=1}^n \in \bigtimes_{j=1}^n A_{j,e} \mid k \in \mathbb{N}, \forall i \in [k] : (a_{i1}, \dots, a_{in}) \in R_{e'}, \forall j \in [n] : \sum_{i=1}^k m_{ij} \in C\}.$$

- For all weighted types  $e = M_C^{e'}$ ,

$$R_e = \{(M_C^{\pi_j}(f))_{j=1}^n \in \bigtimes_{j=1}^n M_C^{A_j} \mid f \in M_C^{R_{e'}}\} \quad (5)$$

where  $\pi_j : R_s \rightarrow A_j$  maps the  $n$ -tuples of  $R_{e'}$  to their  $j$ -th components.

Hence for all  $(f_1, \dots, f_n) \in \bigtimes_{j=1}^n M_C^{A_j}$ ,

$$(f_1, \dots, f_n) \in R_e \Leftrightarrow \exists f \in M_C^{R_{e'}} : \forall 1 \leq j \leq n : M_C^{\pi_j}(f) = f_j$$

(see Preliminaries).

(5) generalizes the definition given in [89], section 2.1, [91], Example 2.9, and [77], section 4.5, from binary to  $n$ -ary relations.

In particular, if  $n = 2$ ,  $M = 2$  and  $e = M^s$ , then  $M_M^{R_s} = 2_\omega^{R_s}$  and thus by (5), for all  $(f_1, f_2) \in 2_\omega^{A_{1,s}} \times 2_\omega^{A_{2,s}}$ ,

$$\begin{aligned}
(f_1, f_2) \in R_e &\Leftrightarrow \exists f \in 2_{\omega}^{R_s} : \forall i \in \{1, 2\} : 2_{\omega}^{\pi_i}(f) = f_i \\
&\Leftrightarrow \exists f \in 2_{\omega}^{R_s} : \forall i \in \{1, 2\} : \lambda a. \sum \{f(a_1, a_2) \mid \pi_i(a_1, a_2) = a\} = f_i \\
&\Leftrightarrow \exists f \in 2_{\omega}^{R_s} : \lambda a_1. \sum \{f(a_1, a_2) \mid (a_1, a_2) \in R_s\} = f_1 \wedge \\
&\quad \lambda a_2. \sum \{f(a_1, a_2) \mid (a_1, a_2) \in R_s\} = f_2 \\
&\Leftrightarrow \exists f \in 2_{\omega}^{R_s} : \forall a_1 \in A_{1,s} : \sum \{f(a_1, a_2) \mid (a_1, a_2) \in R_s\} = f_1(a_1) \wedge \\
&\quad \forall a_2 \in A_{2,s} : \sum \{f(a_1, a_2) \mid (a_1, a_2) \in R_s\} = f_2(a_2). \quad (6)
\end{aligned}$$

In terms of the powerset functor  $\mathcal{P}$ , (6) reads as follows:

For all  $(as_1, as_2) \in \mathcal{P}(A_{1,s}) \times \mathcal{P}(A_{2,s})$ ,

$$\begin{aligned}
(as_1, as_2) \in R_e &\Leftrightarrow \exists R' \in \mathcal{P}(R_s) : \\
&\quad \forall a_1 \in A_{1,s} : (\exists a_2 \in A_{2,s} : (a_1, a_2) \in R' \Leftrightarrow a_1 \in as_1) \wedge \\
&\quad \forall a_2 \in A_{2,s} : (\exists a_1 \in A_{1,s} : (a_1, a_2) \in R' \Leftrightarrow a_2 \in as_2) \\
&\Leftrightarrow \forall a_1 \in as_1 \exists a_2 \in as_2 : (a_1, a_2) \in R_s \wedge \\
&\quad \forall a_2 \in as_2 \exists a_1 \in as_1 : (a_1, a_2) \in R_s.
\end{aligned}$$

## Exercise 9

Show by induction on  $e$  that for all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $R_e$  is a subset of  $A_{e,1} \times \dots \times A_{e,n}$ , and thus  $R$  is indeed a  $\mathcal{T}_q(S, \mathcal{I})$ -sorted  $n$ -ary relation on  $A$ .  $\square$

The relations occurring in universal algebra mostly have arity 1 (like the image of a function) or 2 (like the kernel of a function; see [Preliminaries](#)). The above-defined lifting of  $S$ -sorted to  $\mathcal{T}_q(S, \mathcal{I})$ -sorted relations transfers images, kernels and other algebraic relations from  $\text{Set}^S$  to  $\text{Mod}(S, \mathcal{I})$ .

## Lemma LIFT

Let  $g, h \in \text{Mod}(S, \mathcal{I})(A, B)$ ,  $R$  be an  $S$ -sorted subset of  $A$  and  $R'$  be an  $S$ -sorted binary relation on  $B$  such that for all  $s \in S$ ,

$$R_s = \{a \in A_s \mid g_s(a) = h_s(a)\} \quad \text{and} \quad R'_s = \{(g_s(a), h_s(a)) \mid a \in A_s\}.$$

Then for all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,

$$R_e = \{a \in A_e \mid g_e(a) = h_e(a)\}, \tag{1}$$

$$R'_e = \{(g_e(a), h_e(a)) \mid a \in A_e\}. \tag{2}$$

*Proof.* Induction on the size of  $e$ . In category-theoretic terms, (1) and (2) tell us that the functor  $F_e$  preserves equalizers and coequalizers, respectively.  $\square$

## Signatures

Given  $e, e' \in \mathcal{T}_q(S, \mathcal{I})$ , a typed symbol of the form  $f : e \rightarrow e'$  is called **an arrow over**  $\mathcal{T}_q(S, \mathcal{I})$ .  $\text{src}(f) = e$  and  $\text{trg}(f) = e'$  are the **source** and **target of**  $f$ , respectively.

A **signature**  $\Sigma = (S, \mathcal{I}, F)$  consists of a finite sets  $S, \mathcal{I}, F$  of sorts, an alphabet  $\mathcal{I}$  and arrows over  $\mathcal{T}_q(S, \mathcal{I})$ , respectively. Hence  $\Sigma$  can be regarded as a finite graph with types from  $\mathcal{T}_q(S, \mathcal{I})$  as nodes and arrows over  $\mathcal{T}_q(S, \mathcal{I})$  as edges.

Given  $n > 0$  and  $s_1, \dots, s_n \in S$ ,  $f : s_1 \times \dots \times s_n \rightarrow 2 \in F$  is called an  **$n$ -ary predicate**.

Given  $s \in S$ ,  $f : e \rightarrow s \in F$  is a **constructor** (for  $s$ ).

Given  $s \in S$ ,  $f : e \rightarrow M_C^s \in F$  is a **weighted constructor** (for  $s$ ).

Given  $s \in S$ ,  $f : s \rightarrow e \in F$  is a **destructor** (for  $s$ ).

$\Sigma$  is **constructive** (**destructive**) if  $F$  consists of (weighted) constructors (destructors).

A constructive (destructive) signature  $\Sigma = (S, \mathcal{I}, F)$  is **finitary** if for all  $c : e \rightarrow s \in F$  ( $d : s \rightarrow e \in F$ ),  $e = s_1 \times \dots \times s_n$  ( $e = s_1 + \dots + s_n$ ) for some  $n > 0$  and  $s_1, \dots, s_n \in S \cup \mathcal{I}$ .

$\Sigma$  is **polynomial** if for all  $f : e \rightarrow e' \in F$ ,  $e$  and  $e'$  are polynomial types.

Let  $\Sigma = (S, \mathcal{I}, F)$  and  $\Sigma' = (S', \mathcal{I}', F')$ .

$\Sigma \cup \Sigma'$  denotes the componentwise union of  $\Sigma$  and  $\Sigma'$ .

$\Sigma$  is a **subsignature** of  $\Sigma'$  and  $\Sigma'$  is an **extension** of  $\Sigma$  if  $\Sigma'$  includes  $\Sigma$  componentwise.

A **signature morphism**  $\sigma : \Sigma \rightarrow \Sigma'$  is a mapping from  $S \cup \mathcal{I} \cup F$  to  $S' \cup \mathcal{I}' \cup F'$  such that  $\sigma(S \cup \mathcal{I}) \subseteq S' \cup \mathcal{I}'$  and for all  $f : e \rightarrow e' \in F$ ,  $\sigma(f) : \sigma^*(e) \rightarrow \sigma^*(e') \in F'$  where  $\sigma^*(e)$  denotes the type obtained from  $e$  by replacing every  $s \in S \cup \mathcal{I}$  with  $\sigma(s)$ .

We also write  $\Sigma[s'_i/s_i \mid 1 \leq i \leq n]$  if  $\{s_1, \dots, s_n\} = \{s \in S \cup \mathcal{I} \cup F \mid \sigma(s) \neq s\}$  and for all  $1 \leq i \leq n$ ,  $\sigma(s_i) = s'_i$ .

## $\Sigma$ -arrows

Let  $\Sigma = (S, \mathcal{I}, F)$ ,  $AT$  be the set of arrow types of  $\mathcal{T}(S, \mathcal{I})$  (see chapter 7) and  $V$  be a set of  $AT$ -sorted variables. The  $AT$ -sorted set  $Arr_{\Sigma}(V)$  of  **$\Sigma$ -arrows over  $V$**  is the least set of arrows over  $\mathcal{T}_q(S, \mathcal{I})$  that contains  $F$  and satisfies the following conditions:

- $F \cup V \subseteq Arr_{\Sigma}(V)$ .
- For all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $\text{id}_e : e \rightarrow e \in Arr_{\Sigma}(V)$ . (identities)
- For all  $f : e \rightarrow e', g : e' \rightarrow e'' \in Arr_{\Sigma}(V)$ ,  $g \circ f : e \rightarrow e'' \in Arr_{\Sigma}(V)$ . (composition)
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_q(S, \mathcal{I})$ ,  $i \in I$  and  $(f_i : e_i \rightarrow e')_{i \in I} \in Arr_{\Sigma}(V)^I$ ,  
 $\iota_i : e_i \rightarrow e$ ,  $[f_i]_{i \in I} : e \rightarrow e' \in Arr_{\Sigma}(V)$ . (injections and sum extensions)

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_q(S, \mathcal{I})$ ,  $i \in I$  and  $(f_i : e' \rightarrow e_i)_{i \in I} \in \text{Arr}_{\Sigma}(V)^I$ ,  
 $\pi_i : e \rightarrow e_i$ ,  $\langle f_i \rangle_{i \in I} : e' \rightarrow e \in \text{Arr}_{\Sigma}(V)$ . (projections and product extensions)
- For all  $e, e' \in \mathcal{T}_q(S, \mathcal{I})$  and natural transformations  $\tau : F_e \rightarrow F_{e'}$ ,  
 $\bar{\tau} : e \rightarrow e' \in \text{Arr}_{\Sigma}(V)$ . (type transformations)

Let  $I \subseteq \mathcal{I}$  and  $\text{fork}_e : F_{e \times 2} \rightarrow F_{e+e}$  be defined as follows: For all  $A \in \text{Mod}(S, \mathcal{I})$  and  $a \in A_e \times 2$ ,

$$\text{fork}_{e,A}(a) = \begin{cases} \iota_1(\pi_1(a)) & \text{if } \pi_2(a) = 1, \\ \iota_2(\pi_1(a)) & \text{if } \pi_2(a) = 0. \end{cases}$$

## Derived $\Sigma$ -arrows

- For all  $(f_i : e_i \rightarrow e'_i)_{i \in I} \in \text{Arr}_{\Sigma}(V)^I$ , (sums and products)

$$\coprod_{i \in I} f_i = [\iota_i \circ f_i]_{i \in I} : \coprod_{i \in I} e_i \rightarrow \coprod_{i \in I} e'_i \quad \text{and} \quad \prod_{i \in I} f_i = \langle f_i \circ \pi_i \rangle_{i \in I} : \prod_{i \in I} e_i \rightarrow \prod_{i \in I} e'_i.$$

- For all  $e, e' \in \mathcal{T}_q(S, \mathcal{I})$ ,  $p : e \rightarrow 2$ ,  $f, g : e \rightarrow e' \in \text{Arr}_{\Sigma}(V)$ , (tests and conditionals)

$$p? = \overline{\text{fork}_e} \circ \langle id_e, p \rangle : e \rightarrow e + e \quad \text{and} \quad \text{ite}(p, f, g) = [f, g] \circ p? : e \rightarrow e'.$$

The latter is inspired by [30], section 3.4.

The  **$\Sigma$ -arrow congruence**  $\sim_\Sigma$  is the least equivalence relation on  $Arr_\Sigma(V)$  such that the following conditions hold true: Let  $I \subseteq \mathcal{I}$ .

- For all  $f : e_1 \rightarrow e_2, g : e_2 \rightarrow e_3, h : e_3 \rightarrow e_4 \in Arr_\Sigma(V)$ ,  $h \circ (g \circ f) \sim_\Sigma (h \circ g) \circ f$ . (1)

- For all  $f : e \rightarrow e' \in Arr_\Sigma(V)$ ,  $f \circ id_e \sim_\Sigma f$  and  $id_{e'} \circ f \sim_\Sigma f$ . (2)

- For all  $(f_i : e_i \rightarrow e)_{i \in I} \in Arr_\Sigma(V)^I$  and  $f : \coprod_{i \in I} e_i \rightarrow e \in Arr_\Sigma(V)$ ,

$$[f_i]_{i \in I} \circ \iota_i \sim_\Sigma f_i \quad \text{and} \quad [f \circ \iota_i]_{i \in I} \sim_\Sigma f. \quad (3)$$

- For all  $(f_i : e \rightarrow e_i)_{i \in I} \in Arr_\Sigma(V)^I$  and  $f : e \rightarrow \prod_{i \in I} e_i \in Arr_\Sigma(V)$ ,

$$\pi_i \circ \langle f_i \rangle_{i \in I} \sim_\Sigma f_i \quad \text{and} \quad \langle \pi_i \circ f \rangle_{i \in I} \sim_\Sigma f. \quad (4)$$

- For all  $f_1, g_1 : e \rightarrow e' \in Arr_\Sigma(V)$  and  $f_2, g_2 : e' \rightarrow e'' \in Arr_\Sigma(V)$ ,

$$f_1 \sim_\Sigma g_1 \quad \text{and} \quad f_2 \sim_\Sigma g_2 \quad \text{imply} \quad f_2 \circ f_1 \sim_\Sigma g_2 \circ g_1.$$

- For all  $(f_i : e_i \rightarrow e)_{i \in I}, (g_i : e_i \rightarrow e)_{i \in I} \in Arr_\Sigma(V)^I$ ,

$$\forall i \in I : f_i \sim_\Sigma g_i \quad \text{implies} \quad [f_i]_{i \in I} \sim_\Sigma [g_i]_{i \in I}.$$

- For all  $(f_i : e \rightarrow e_i)_{i \in I}, (g_i : e \rightarrow e_i)_{i \in I} \in Arr_\Sigma(V)^I$ ,

$$\forall i \in I : f_i \sim_\Sigma g_i \quad \text{implies} \quad \langle f_i \rangle_{i \in I} \sim_\Sigma \langle g_i \rangle_{i \in I}.$$

$t \in Arr_{\Sigma}(V)$  is **ground** if  $t$  does not contain variables.  $Arr_{\Sigma}$  denotes the set of ground elements of  $Arr_{\Sigma}(V)$ .

The quotient

$$Mor(\Sigma) =_{def} Arr_{\Sigma}/\sim_{\Sigma}$$

is the  $\mathcal{T}_q(S, \mathcal{I})$ -sorted set of morphisms of the **free or syntactic category**  $\mathcal{K}(\Sigma)$  over  $\Sigma$ .

The objects of  $\mathcal{K}(\Sigma)$  are the quantitative types over  $(S, \mathcal{I})$ . Compositions, extensions and forks of  $\mathcal{K}(\Sigma)$  are defined as liftings of the synonymous operator symbols of  $Arr_{\Sigma}$  to functions on  $\sim_{\Sigma}$ -equivalence classes.

Hence by (1) and (2),  $\mathcal{K}(\Sigma)$  is a category. By (3), (4) and the characterization of sums and products given by equations (10), (11), (21) and (22) in chapter 2,  $\mathcal{K}(\Sigma)$  has sums and products.

In the following signatures  $\Sigma = (S, \mathcal{I}, F)$ ,  $\mathcal{I}$  is given by the union over all primitive types occurring in the source or target type of arrows of  $F$  and  $\curvearrowleft$  points to the carrier of an initial or final  $\Sigma$ -algebra (see chapter 9).

Many examples presented in the sample sections of the present chapter as well as of chapters 9, 13, 14 and 23 have been implemented and tested in Haskell (see the modules Coalg.hs and Compiler.hs that are part of the Painter tool).

## Sample constructive signatures

Let  $X, Y, Act$  be nonempty sets  $\neq 1$  and  $(M, +, 0)$  be a commutative monoid.

- $Mon$  (nonempty unlabelled binary trees; e.g., monoids)

$$S = \{mon\}, \quad F = \{one : 1 \rightarrow mon, \ mul : mon \times mon \rightarrow mon\}.$$

- $Nat \bowtie \mathbb{N}$

$$S = \{nat\}, \quad F = \{zero : 1 \rightarrow nat, \ succ : nat \rightarrow nat\}.$$

- $Dyn(X, Y) \bowtie X^* \times Y$  (dynamics;  $Y$ -pointed automata)

$$S = \{state\}, \quad F = \{cons : X \times state \rightarrow state, \ \alpha : Y \rightarrow state\}.$$

$$\text{List}(X) =_{def} \text{Dyn}(X, 1) \bowtie X^*$$

$\text{List}(X)$  is equivalent to  $(\{\text{state}\}, X \cup 1, \{\text{list} : X^* \rightarrow \text{state}\})$  (see chapter 13).

$\text{List}(1)$  is equivalent to  $\text{Nat}$ .

$$\text{Nelist}(X) =_{def} \text{Dyn}(X, X). \bowtie X^+$$

$\text{Nelist}(X)$  is equivalent to  $(\{\text{state}\}, X \cup 1, , \{\text{list} : X^+ \rightarrow \text{state}\})$ .

- $\text{WDyn}(X, Y, M)$  (pointed  $M$ -weighted automata)

$$S = \{\text{state}\}, \quad F = \{\text{cons} : X \times \text{state} \rightarrow M_\omega^{\text{state}}, \alpha : Y \rightarrow \text{state}\}.$$

- $\text{coStream}(X) \bowtie X^{\mathbb{N}}$  (semiautomata)

$$S = \{\text{state}\}, \quad F = \{\text{cons} : X \times \text{state} \rightarrow \text{state}\}.$$

- $\text{WcoStream}(X, M)$  ( $M$ -weighted semiautomata)

$$S = \{\text{state}\}, \quad F = \{\text{cons} : X \times \text{state} \rightarrow M^{\text{state}}\}.$$

- $coDAut(X, Y) \bowtie Y^{X^*}$  (Moore automata with input from  $X$  and output from  $Y$ )

$$S = \{state\}, \quad F = \{new : state^X \times Y \rightarrow state\}.$$

- $coDAut(X, 2) \bowtie \mathcal{P}(X^*)$  (deterministic acceptors of words over  $X$ )
- $Bintree(X) \bowtie ftr(2, X)$  (binary trees of finite depth with node labels from  $X$ )

$$S = \{btree\}, \quad F = \{ \begin{aligned} & bjoin : X \times btree \times btree \rightarrow btree, \\ & empty : 1 \rightarrow btree \end{aligned} \}.$$

- $Nebintree(X) \bowtie ftr(2, X) \setminus \{\Omega\}$

(nonempty binary trees of finite depth with node labels from  $X$ )

$$S = \{btree\}, \quad F = \{ \begin{aligned} & bjoin : X \times btree \times btree \rightarrow btree, \\ & ljoin, rjoin : X \times btree \rightarrow btree, \\ & leaf : X \rightarrow btree \end{aligned} \}.$$

- $Nebintree2(X) \Leftrightarrow \{t \in ftr(2, X) \setminus \{\Omega\} \mid \forall w \in 2^* : w0, w1 \in def(t) \vee w0, w1 \notin def(t)\}$  (nonempty binary trees of finite depth with node labels from  $X$  and outdegree 0 or 2)

$$S = \{btree\}, \quad F = \{ \begin{aligned} & bjoin : X \times btree \times btree \rightarrow btree, \\ & leaf : X \rightarrow btree \end{aligned} \}.$$

- $Tree(X) \Leftrightarrow otr(\mathbb{N}, <, X) \cap ftr(\mathbb{N}, X)$  (finitely branching trees of finite depth with node labels from  $X$ )

$$S = \{tree, trees\}, \quad F = \{ \begin{aligned} & join : X \times trees \rightarrow tree, \\ & cons : tree \times trees \rightarrow trees, \quad nil : 1 \rightarrow trees \end{aligned} \}.$$

- $Tree_\omega(X) \Leftrightarrow otr(\mathbb{N}, <, X) \cap wtr(\mathbb{N}, X)$  (finitely or infinitely branching trees of finite depth with node labels from  $X$ )

$$S = \{tree\}, \quad F = \{join : X \times tree^\infty \rightarrow tree\}.$$

- $ETree(X, Y) \Leftrightarrow ftr(Y, X)$  (finitely branching trees of finite depth with node labels from  $X$  and edge labels from  $Y$ )

$$S = \{tree\}, \quad F = \{join : X \times (Y \times tree)^* \rightarrow tree\}.$$

- $Reg(X) = (S, \mathcal{P}(X) \cup 1, F)$   $\bowtie$  regular expressions over  $X$  with  $() \notin X$

$$\begin{aligned}
 S &= \{ \text{state} \}, \\
 F &= \{ \text{par} : \text{state} \times \text{state} \rightarrow \text{state}, && \text{(parallel composition)} \\
 &\quad \text{seq} : \text{state} \times \text{state} \rightarrow \text{state}, && \text{(sequential composition)} \\
 &\quad \text{iter} : \text{state} \rightarrow \text{state}, && \text{(iteration)} \\
 &\quad \text{eps} : 1 \rightarrow \text{state}, && \text{(empty word)} \\
 &\quad \text{base} : \mathcal{P}(X) \rightarrow \text{state} \} && \text{(base languages)}
 \end{aligned}$$

- $CCS(Act)$  ↣ Calculus of Communicating Systems (see section 26.2)

$$\begin{aligned}
 S &= \{ \text{proc} \}, \\
 F &= \{ \text{pre} : Act \times \text{proc} \rightarrow \text{proc}, && \text{(prefixing by an action)} \\
 &\quad \text{cho} : \text{proc} \times \text{proc} \rightarrow \text{proc}, && \text{(choice)} \\
 &\quad \text{par} : \text{proc} \times \text{proc} \rightarrow \text{proc}, && \text{(parallelism)} \\
 &\quad \text{res} : \text{proc} \times Act \rightarrow \text{proc}, && \text{(restriction)} \\
 &\quad \text{rel} : \text{proc} \times Act^{Act} \rightarrow \text{proc} \}.
 \end{aligned}$$

## Sample destructive signatures

Let  $X, Y, Act$  be nonempty sets and  $(M, +, 0)$  be a commutative monoid.

- $coNat$  ↣  $\mathbb{N} \cup \{\omega\} \cong ltr(1, 1)$  (see chapter 2)

$$S = \{nat\}, \quad F = \{pred : nat \rightarrow 1 + nat\}.$$

- $Stream(X)$  ↣  $X^{\mathbb{N}}$

$$S = \{state\}, \quad F = \{head : state \rightarrow X, \ tail : state \rightarrow state\}.$$

- $WStream(X, M)$

$$S = \{state\}, \quad F = \{head : state \rightarrow X, \ tail : state \rightarrow M^{state}\}.$$

- $WStream^*(X, M) \Leftrightarrow otr(\mathbb{N} \times M, (<, \Delta_M), X)$

$$S = \{state\}, \quad F = \{head : state \rightarrow X, \ tail : state \rightarrow (state \times M)^*\}.$$

- $coDyn(X, Y) \Leftrightarrow X^* \times Y \cup X^{\mathbb{N}}$  (codynamics)

$$S = \{state\}, \quad F = \{split : state \rightarrow X \times state + Y\}.$$

$$coList(X) =_{def} coDyn(X, 1) \Leftrightarrow X^* \cup X^{\mathbb{N}}$$

$coList(1)$  is equivalent to  $coNat$ .

$$coNelist(X) =_{def} coDyn(X, X) \Leftrightarrow X^+ \cup X^{\mathbb{N}}$$

- $\text{infBintree}(X) \rightsquigarrow X^{2^*}$  (binary trees of infinite depth with node labels from  $X$ )

$$S = \{btree\}, \quad F = \{left, right : btree \rightarrow btree, \ root : btree \rightarrow X\}.$$

- $\text{coBintree}(X) \rightsquigarrow ltr(2, X)$

(binary trees of finite or infinite depth with node labels from  $X$ )

$$S = \{btree\}, \quad F = \{split : btree \rightarrow X \times btree \times btree + 1\}.$$

- $\text{coNebintree}(X) \rightsquigarrow ltr(2, X) \setminus \{\Omega\}$

(nonempty binary trees of finite or infinite depth with node labels from  $X$ )

$$S = \{btree\},$$

$$F = \{split : btree \rightarrow X \times btree \times btree + X \times btree + X \times btree + X\}.$$

- $\text{coNebintree2}(X)$

$$\rightsquigarrow \{t \in ltr(2, X) \setminus \{\Omega\} \mid \forall w \in 2^* : w0, w1 \in \text{def}(t) \vee w0, w1 \notin \text{def}(t)\}$$

(nonempty binary trees of finite or infinite depth with node labels from  $X$  and out-degree 0 or 2)

$$S = \{btree\}, \quad F = \{split : btree \rightarrow X \times btree \times btree + X\}.$$

- $\text{coTree}(X) \Leftrightarrow \text{otr}(\mathbb{N}, <, X)$  (finitely or infinitely branching trees of finite or infinite depth with node labels from  $X$ )

$$\begin{aligned} S &= \{ \text{ tree, trees } \}, \\ F &= \{ \text{ subtrees} : \text{tree} \rightarrow \text{trees}, \text{ root} : \text{tree} \rightarrow X, \\ &\quad \text{split} : \text{trees} \rightarrow \text{tree} \times \text{trees} + 1 \ }. \end{aligned}$$

- $\text{coTree}_\omega(X) \Leftrightarrow \text{otr}(\mathbb{N}, <, X) \cap \text{fbtr}(\mathbb{N}, X)$  (finitely branching trees of finite or infinite depth with node labels from  $X$ )

$$\begin{aligned} S &= \{ \text{ tree } \}, \\ F &= \{ \text{ subtrees} : \text{tree} \rightarrow \text{tree}^*, \\ &\quad \text{root} : \text{tree} \rightarrow X \ }. \end{aligned}$$

- $\text{infTree}(X) \Leftrightarrow \text{otr}(\mathbb{N}, <, X) \cap \text{fbtr}(\mathbb{N}, X) \cap \text{itr}(\mathbb{N}, X)$   
(finitely branching trees of infinite depth with node labels from  $X$ )

$$\begin{aligned} S &= \{ \text{ tree } \}, \\ F &= \{ \text{ subtrees} : \text{tree} \rightarrow \text{tree}^+, \\ &\quad \text{root} : \text{tree} \rightarrow X \ }. \end{aligned}$$

- $coETree(X, Y)$  (finitely or infinitely branching trees of finite or infinite depth with node labels from  $X$  and edge labels from  $Y$ )

$$\begin{aligned} S &= \{ \text{ tree, trees } \}, \\ F &= \{ \text{ subtrees} : \text{tree} \rightarrow \text{trees}, \text{ root} : \text{tree} \rightarrow X, \\ &\quad \text{split} : \text{trees} \rightarrow Y \times \text{tree} \times \text{trees} + 1 \}. \end{aligned}$$

- $Proctree(Act)$  (process trees whose edges are labelled with actions; see section 26.2)

$$S = \{ \text{tree} \}, \quad F = \{ \delta : \text{tree} \rightarrow (Act \times \text{tree})^* \}.$$

- $Med(X)$  (deterministic Medvedev automata; semiautomata)

$$S = \{ \text{state} \}, \quad F = \{ \delta : \text{state} \rightarrow \text{state}^X \}.$$

$Med(1)$  is equivalent to  $coStream(1)$ .

- $NMed(X)$  (nondeterministic Medvedev automata)

$$S = \{ \text{state} \}, \quad F = \{ \delta : \text{state} \rightarrow (2^{\text{state}})^X \}.$$

- $NMed^*(X) \Leftrightarrow otr(X \times \mathbb{N}, (\Delta_X, <), 1)$

$$S = \{ \text{state} \}, \quad F = \{ \delta : \text{state} \rightarrow (\text{state}^*)^X \}.$$

- $WMed(X, M)$  ( $M$ -weighted Medvedev automata)

$$S = \{state\}, \quad F = \{\delta : state \rightarrow (M^{state})^X\}.$$

- $WMed^*(X, M) \bowtie otr(X \times \mathbb{N} \times M, (\Delta_X, <, \Delta_M), 1)$

$$S = \{state\}, \quad F = \{\delta : state \rightarrow ((M \times state)^*)^X\}.$$

- $DAut(X, Y) \bowtie Y^{X^*}$  (Moore automata with input from  $X$  and output from  $Y$ ;  $Y$ -colored automata)

$$S = \{state\}, \quad F = \{\delta : state \rightarrow state^X, \beta : state \rightarrow Y\}.$$

$DAut(1, X)$  is equivalent to  $Stream(X)$ .

$DAut(2, X)$  is equivalent to  $infBintree(X)$ .

$Acc(X) =_{def} DAut(X, 2) \bowtie \mathcal{P}(X^*)$  (deterministic acceptors of words over  $X$ )

- $Mealy(X, Y) \bowtie Y^{X^+}$  (Mealy automata)

$$S = \{state\}, \quad F = \{\delta : state \rightarrow state^X, \beta : state \rightarrow Y^X\}.$$

- $PAut(X, Y) \Leftrightarrow ltr(X, Y)$  (partial automata with input from  $X$  and output from  $Y$ )

$$S = \{state\}, \quad F = \{\delta : state \rightarrow (1 + state)^X, \beta : state \rightarrow Y\}.$$

$PAut(1, Y)$  is equivalent to  $coNelist(Y)$  because for all sets  $A$ ,

$$(1 + A)^1 \times Y \cong (1 + A) \times Y \cong 1 \times Y + A \times Y \cong Y + A \times Y \cong Y \times A + Y.$$

$PAut(2, Y)$  is equivalent to  $coNebintree(Y)$  because for all sets  $A$ ,

$$\begin{aligned} (1 + A)^2 \times Y &\cong (1 + A + A + A^2) \times Y \cong 1 \times Y + A \times Y + A \times Y + A^2 \times Y \\ &\cong A \times Y \times A + A \times Y + A \times Y + Y. \end{aligned}$$

- $NAut(X, Y)$  (nondeterministic automata with input from  $X$  and output from  $Y$ )

$$S = \{state\},$$

$$F = \{\delta : state \rightarrow \mathcal{P}_\omega(state)^X, \beta : state \rightarrow Y\}.$$

$NAcc(X) =_{def} NAut(X, 2) \Leftrightarrow \mathcal{P}(X^*)$  (non-deterministic acceptors of words over  $X$ )

$NAut_{\times}(X, Y)$

$$S = \{state\},$$

$$F = \{\delta : state \rightarrow \mathcal{P}_{\omega}(X \times state), \beta : state \rightarrow Y\}.$$

$NAut^*(X, Y) \rightsquigarrow otr(X \times \mathbb{N}, (\Delta_X, <), Y)$

$$S = \{state\},$$

$$F = \{\delta : state \rightarrow (state^*)^X, \beta : state \rightarrow Y\}.$$

$NAut_{\times}^*(X, Y)$

$$S = \{state\},$$

$$F = \{\delta : state \rightarrow (X \times state)^*, \beta : state \rightarrow Y\}.$$

By (3) and (4) in chapter 7 (for  $e = 1$  and  $I = X$ ),  $2^{X \times state}$  and  $(2^{state})^X$  are equivalent types and thus  $NAut_{\times}(X, Y)$  and  $NAut(X, Y)$  are equivalent signatures, while  $NAut_{\times}^*(X, Y)$  is only a quotient of  $NAut^*(X, Y)$ .

- $WAut(X, M, Y)$  (colored  $M$ -weighted automata)

$$S = \{state\}, \quad F = \{\delta : state \rightarrow (M^{state})^X, \beta : state \rightarrow Y\}.$$

$WAut(1, M, X)$  is equivalent to  $WStream(X, M)$ .

- $WAut^*(X, M, Y) \Leftrightarrow otr(X \times \mathbb{N} \times M, (\Delta_X, <, \Delta_M), Y)$

$$S = \{state\}, \quad F = \{\delta : state \rightarrow ((M \times state)^*)^X, \beta : state \rightarrow Y\}.$$

- $PrAut(X, Y)$  (probabilistic automata with input from  $X$  and output from  $Y$ )

$$S = \{state\}, \quad F = \{\delta : state \rightarrow \mathcal{D}(state)^X, \beta : state \rightarrow Y\}.$$

- Let  $\Sigma = (S, \mathcal{I}, C)$  be a finitary signature (see above).

$TAcc(\Sigma) \Leftrightarrow \mathcal{P}(T_\Sigma)$  (deterministic top-down tree acceptors; see  $\Sigma$ -terms and -coterminals)

$$F = \{\delta_c : s \rightarrow s'_1 \times \cdots \times s'_n \mid c : s_1 \times \cdots \times s_n \rightarrow s \in C\}$$

where for all  $s \in S \cup \mathcal{I}$ ,  $s' =_{def} \text{if } s \in S \text{ then } s \text{ else } \mathcal{P}(s)$ .

If  $S$  and  $\mathcal{I}$  are singletons, say  $S = \{state\}$  and  $\mathcal{I} = \{Y\}$ , and for all  $c : e \rightarrow s \in C$ ,  $e \in \{state, Y\}$ , then  $TAcc(\Sigma)$  is equivalent to  $Mealy(C', \mathcal{P}(Y))$  where

$$C' = \{c : e \rightarrow s \in C \mid e = state\}.$$

$NTAcc(\Sigma) \Leftrightarrow \mathcal{P}(T_\Sigma)$  (nondeterministic top-down tree acceptors)

$$F = \{\delta_c : s \rightarrow \mathcal{P}_\omega(e) \mid c : e \rightarrow s \in C\}.$$

$NTAcc^*(\Sigma)$  (polynomial nondeterministic top-down tree acceptors)

$$F = \{\delta_c : s \rightarrow e^* \mid c : e \rightarrow s \in C\}.$$

- Let  $X_1, \dots, X_n, Y_1, \dots, Y_n, E_1, \dots, E_n$  be nonempty sets and

$$\textcolor{brown}{BS} = \{X_1, \dots, X_n, Y_1, \dots, Y_n, E_1, \dots, E_n\}.$$

$Class(\textcolor{brown}{BS})$  (object classes with  $n$  methods [90])

$$S = \{state\}, \quad F = \{m_i : state \rightarrow ((Y_i \times state) + E_i)^{X_i} \mid 1 \leq i \leq n\}.$$

- UML diagrams (object class diagrams with  $n$  classes and associations between them)

$$S = \{s_1, \dots, s_n\}, \quad F = \{assoc_i : s_i \rightarrow s_{k_1} \times \dots \times s_{k_i} \mid 1 \leq i \leq n, 1 \leq k_j \leq n\}.$$

- $Graph(X, Y)$  (node- and edge-labelled graphs)

$$S = \{node, edge\},$$

$$F = \{source, target : edge \rightarrow node, nlabel : node \rightarrow X, elabel : edge \rightarrow Y\}.$$

## $\Sigma$ -algebras

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature.

A functor  $\mathcal{A} : \mathcal{K}(\Sigma) \rightarrow \text{Set}$  is a  **$\Sigma$ -algebra** if the  $\mathcal{T}_q(S, \mathcal{I})$ -sorted set  $\mathcal{A} = (\mathcal{A}(e))_{e \in \mathcal{T}_q(S, \mathcal{I})}$ , called the **carrier of  $\mathcal{A}$** , is a quantitative model over  $(S, \mathcal{I})$  and  $\mathcal{A}$  preserves injections, projections, type transformations and sum and product extensions, i.e.,  $\mathcal{A}$  interprets the morphisms of  $\mathcal{K}(\Sigma)$  inductively as follows:

Let  $I \subseteq \mathcal{I}$ .

- For all identities  $\text{id}_e$  of  $\text{Mor}(\Sigma)$ ,  $\mathcal{A}(\text{id}_e) = \text{id}_{\mathcal{A}_e}$ .
- For all  $i \in I$  and injections  $\iota_i$  of  $\text{Mor}(\Sigma)$ ,  $\mathcal{A}(\iota_i) = \iota_i$ .
- For all  $i \in I$  and projections  $\pi_i$  of  $\text{Mor}(\Sigma)$ ,  $\mathcal{A}(\pi_i) = \pi_i$ .
- For all  $e, e' \in \mathcal{T}_q(S, \mathcal{I})$  and natural transformations  $\tau : F_e \rightarrow F_{e'}$ ,  $\mathcal{A}(\bar{\tau}) = \tau_A$ .
- For all  $f : e \rightarrow e'$ ,  $g : e' \rightarrow e'' \in \text{Mor}(\Sigma)$ ,  $\mathcal{A}(g \circ f) = \mathcal{A}(g) \circ \mathcal{A}(f)$ .
- For all  $(f_i : e_i \rightarrow e)_{i \in I} \in \text{Mor}(\Sigma)^I$ ,  $\mathcal{A}([f_i]_{i \in I}) = [\mathcal{A}(f_i)]_{i \in I}$ .
- For all  $(f_i : e \rightarrow e_i)_{i \in I} \in \text{Mor}(\Sigma)^I$ ,  $\mathcal{A}(\langle f_i \rangle_{i \in I}) = \langle \mathcal{A}(f_i) \rangle_{i \in I}$ .

See the Preliminaries for the functions and operators on the right-hand sides of these equations.

## Exercise 10

Show that every algebra  $\mathcal{A}$  interprets derived arrows in the respectively desired way, e.g.,

- for all  $(f_i : e_i \rightarrow e'_i)_{i \in I} \in \text{Mor}(\Sigma)^I$ ,

$$\begin{aligned}\mathcal{A}(\coprod_{i \in I} f_i) &= [\iota_i \circ \mathcal{A}(f_i)]_{i \in I}, \\ \mathcal{A}(\prod_{i \in I} f_i) &= \langle \mathcal{A}(f_i) \circ \pi_i \rangle_{i \in I},\end{aligned}$$

- for all  $f = (f_s : e_s \rightarrow e'_s)_{s \in S} \in \text{Mor}(\Sigma)^I$  and  $(e_i)_{i \in I} \in \mathcal{T}_q(S, \mathcal{I})^I$ ,

$$\begin{aligned}\mathcal{A}(f_I) &= \text{id}_I, \\ \mathcal{A}(f_{\prod_{i \in I} e_i}) &= \prod_{i \in I} \mathcal{A}(f_{e_i}), \\ \mathcal{A}(f_{\coprod_{i \in I} e_i}) &= \coprod_{i \in I} \mathcal{A}(f_{e_i}). \quad \square\end{aligned}$$

For all  $e \in \mathcal{T}_q(S, \mathcal{I})$  and  $f \in \text{Arr}_{\Sigma}$ ,  $\mathcal{A}(e) = A_e$  and

$$f^{\mathcal{A}} =_{\text{def}} \mathcal{A}([f]_{\sim_{\Sigma}})$$

are called the **interpretations** of  $e$  and  $f$ , respectively, in  $\mathcal{A}$ . Since  $A$  is a model of  $\mathcal{T}_q(S, \mathcal{I})$ ,  $f^{\mathcal{A}}$  is well-defined.

The carrier of  $\mathcal{A}$  is sometimes identified with the union of its components.

For all  $f \in F$ ,  $f^{\mathcal{A}}$  is called an **operation of  $\Sigma$** . For arrows  $f : 1 \rightarrow e$ , we write  $f^{\mathcal{A}}$  instead of  $f^{\mathcal{A}}()$ —provided that ambiguities are excluded.

Given  $f, g \in Arr_{\Sigma}$ ,  $\mathcal{A}$  satisfies the equation  $f = g$ , written as  $\mathcal{A} \models f = g$ , if  $f^{\mathcal{A}} = g^{\mathcal{A}}$ .

The notion of a free or syntactic category and their interpretation by set functors for interpreting signatures has several sources (see, e.g., [137]; [51], Def. 3.7). *Sketches* [24, 25] and *ologs* [158, 159, 160] also provide approaches to modeling in terms of signature graphs and functors mapping its nodes and edges to sets and functions, respectively. In the case of ologs, the graph represents a database schema and the sets and functions the graph is mapped to by a set functor represent tables (relations) and attributes (column indices), respectively.

## Olog example

[160], Example 4.5.2.1, defines a database schema as a quotient category that resembles the free category over a signature. For instance, let  $\Sigma = (S, String, F)$  with

$$S = \{Employee, Department\},$$

and

$$F = \{ \begin{aligned} & \text{manager : } Employee \rightarrow Employee, \\ & \text{worksIn : } Employee \rightarrow Department, \\ & \text{secretary : } Department \rightarrow Employee, \\ & \text{first, last : } Employee \rightarrow String, \\ & \text{name : } Department \rightarrow String \}. \end{aligned}$$

Database constraints are then specified as equations between elements of  $Arr_\Sigma$ , e.g.,

$$\begin{aligned} \text{worksIn} \circ \text{manager} &= \text{worksIn}, \\ \text{worksIn} \circ \text{secretary} &= id_{Department}, \end{aligned}$$

and added to  $\sim_\Sigma$  (see chapter 8). Consequently, database instances satisfying the constraints come as algebra functors on the respective quotient category, which—due to the additional equivalences—might be coarser than our  $\mathcal{K}(\Sigma)$ .  $\square$

## Homomorphisms

Let  $(S, \mathcal{I}, F)$  be a signature and  $\mathcal{A}, \mathcal{B}$  be  $\Sigma$ -algebras with carriers  $A$  and  $B$ , respectively.

A  $Mod(S, \mathcal{I})$ -morphism  $h : A \rightarrow B$  is a  **$\Sigma$ -homomorphism** (or  **$\Sigma$ -homomorphic**) and denoted by  $h : \mathcal{A} \rightarrow \mathcal{B}$  if for all  $f : e \rightarrow e' \in F$ ,

$$h_{e'} \circ f^{\mathcal{A}} = f^{\mathcal{B}} \circ h_e. \quad (1)$$

$h$  is a  **$\Sigma$ -isomorphism** if  $h$  is iso in  $Alg_{\Sigma}$ .

The category of  $\Sigma$ -algebras and  $\Sigma$ -homomorphisms is a subcategory of  $Mod(S, \mathcal{I})$  and denoted by  $Alg_{\Sigma}$ .

For all  $\Sigma$ -homomorphisms  $h : \mathcal{A} \rightarrow \mathcal{B}$ ,

$h$  is epi in  $Alg_{\Sigma}$  iff  $h$  is epi in  $Mod(S, \mathcal{I})$  iff  $h$  is epi in  $Set^S$ .

$h$  is mono in  $Alg_{\Sigma}$  iff  $h$  is mono in  $Mod(S, \mathcal{I})$  iff  $h$  is mono in  $Set^S$ .

$h$  is iso in  $Alg_{\Sigma}$  iff  $h$  is iso in  $Mod(S, \mathcal{I})$  iff  $h$  is iso in  $Set^S$ .

## Lemma HOMARR

Every  $\Sigma$ -homomorphism  $h : \mathcal{A} \rightarrow \mathcal{B}$  is a natural transformation, i.e., the following diagram commutes for all  $f : e \rightarrow e' \in Arr_{\Sigma}$ :

$$\begin{array}{ccc}
 \mathcal{A}(e) & \xrightarrow{h_e} & \mathcal{B}(e) \\
 f^{\mathcal{A}} \downarrow & (4) & \downarrow f^{\mathcal{B}} \\
 \mathcal{A}(e') & \xrightarrow{h_{e'}} & \mathcal{B}(e')
 \end{array}$$

*Proof.* We show (4) by induction on the structure of  $f$ .

If  $f \in F$ , then (4) holds true because  $h$  is  $\Sigma$ -homomorphic.

If  $f = id_e$ , then (4) holds true trivially.

If  $f$  is an injection, say  $f = \iota_i : e_i \rightarrow \coprod_{i \in I} e_i$ , then

$$h_{e'} \circ f^{\mathcal{A}} = h_{\coprod_{i \in I} e_i} \circ \iota_i = \coprod_{i \in I} h_{e_i} \circ \iota_i \stackrel{(18) \text{ in section } 2}{=} \iota_i \circ h_{e_i} = f^{\mathcal{B}} \circ h_e.$$

If  $f$  is a projection, say  $f = \pi_i : \prod_{i \in I} e_i \rightarrow e_i$ , then

$$h_{e'} \circ f^{\mathcal{A}} = h_{e_i} \circ \pi_i \stackrel{(7) \text{ in section } 2}{=} \pi_i \circ \prod_{i \in I} h_{e_i} = \pi_i \circ h_{\prod_{i \in I} e_i} = f^{\mathcal{B}} \circ h_e.$$

If  $f = \bar{\tau}$  for some natural transformation  $\tau : F_e \rightarrow F_{e'}$ , then

$$h_{e'} \circ f^{\mathcal{A}} = F_{e'}(h) \circ \tau_A = \tau_B \circ F_e(h) = f^{\mathcal{B}} \circ h_{e'}.$$

Let  $f = f_2 \circ f_1$  for some  $f_1 : e \rightarrow e'', f_2 : e'' \rightarrow e' \in Arr_{\Sigma}$ . Then (4) holds true for  $f_1, f_2$  by induction hypothesis. Hence

$$\begin{aligned} h_{e'} \circ f^{\mathcal{A}} &= h_{e'} \circ (f_2 \circ f_1)^{\mathcal{A}} = h_{e'} \circ f_2^{\mathcal{A}} \circ f_1^{\mathcal{A}} \stackrel{ind. \text{ hyp.}}{=} f_2^{\mathcal{B}} \circ h_{e''} \circ f_1^{\mathcal{A}} \stackrel{ind. \text{ hyp.}}{=} f_2^{\mathcal{B}} \circ f_1^{\mathcal{B}} \circ h_e \\ &= (f_2 \circ f_1)^{\mathcal{B}} \circ h_e = f^{\mathcal{B}} \circ h_e. \end{aligned}$$

Let  $f = [f_i]_{i \in I}$  for some  $f_i : e_i \rightarrow e'$ ,  $i \in I$ . Then  $e = \coprod_{i \in I} e_i$  and (1) holds true for  $f_i$ ,  $i \in I$ , by induction hypothesis. Hence for all  $i \in I$ ,

$$\begin{aligned} h_{e'} \circ f^A \circ \iota_i &= h_{e'} \circ [f_i]_{i \in I}^A \circ \iota_i = h_{e'} \circ f_i^A \stackrel{\text{ind. hyp.}}{=} f_i^B \circ h_{e_i} = [f_i^B]_{i \in I} \circ \iota_i \circ h_{e_i} \\ (18) \text{ in section 2} \quad &[f_i^B]_{i \in I} \circ \coprod_{i \in I} h_{e_i} \circ \iota_i = [f_i^B]_{i \in I} \circ h_{\coprod_{i \in I} e_i} \circ \iota_i = f^B \circ h_e \circ \iota_i \end{aligned}$$

and thus (4) by (13) in section 2.

Let  $f = \langle f_i \rangle_{i \in I}$  for some  $f_i : e \rightarrow e_i$ ,  $i \in I$ . Then  $e' = \prod_{i \in I} e_i$  and (4) holds true for  $f_i$ ,  $i \in I$ , by induction hypothesis. Hence for all  $i \in I$ ,

$$\begin{aligned} \pi_i \circ h_{e'} \circ f^A &= \pi_i \circ h_{\prod_{i \in I} e_i} \circ \langle f_i \rangle_{i \in I}^A = \pi_i \circ \prod_{i \in I} h_{e_i} \circ \langle f_i^A \rangle_{i \in I} \\ (7) \text{ in section 2} \quad &h_{e_i} \circ \pi_i \circ \langle f_i^A \rangle_{i \in I} = h_{e_i} \circ f_i^A \stackrel{\text{ind. hyp.}}{=} f_i^B \circ h_e = \pi_i \circ \langle f_i^B \rangle_{i \in I} \circ h_e \\ &= \pi_i \circ \langle f_i \rangle_{i \in I}^B \circ h_e = \pi_i \circ f^B \circ h_e \end{aligned}$$

and thus (4) by (2) in section 2. □

A  $\Sigma$ -homomorphism  $h : \mathcal{A} \rightarrow \mathcal{B}$  induces the **image algebra**  $h(\mathcal{A})$ :

- For all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $h(\mathcal{A})(e) = h_e(\mathcal{A}(e))$ .
- For all  $f : e \rightarrow e' \in F$  and  $a \in \mathcal{A}(e)$ ,  $f^{h(\mathcal{A})}(h(a)) = f^B(h(a))$ .

## Algebras and the Yoneda Lemma (see chapter 5)

Applied to  $\Sigma$ -algebras  $\mathcal{A} : \mathcal{K}(\Sigma) \rightarrow Set$ , the **Yoneda Lemma** (see chapter 5) tells us that for all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,

$$Alg_{\Sigma}(\mathcal{R}(e), \mathcal{A}) \cong \mathcal{A}(e) \cong Alg_{\Sigma}(\mathcal{R}'(e), \mathcal{A}), \quad (3)$$

i.e.,  $\mathcal{A}(e)$  is isomorphic to the set of  $\Sigma$ -homomorphisms from the reader functor  $\mathcal{R}(e) = \mathcal{K}(\Sigma)(e, \underline{\phantom{x}})$  to  $\mathcal{A}$  as well as those from the coreader functor  $\mathcal{R}'(e) = \mathcal{K}(\Sigma)(\underline{\phantom{x}}, e)$  to  $\mathcal{A}$ .

As  $\Sigma$ -algebras,  $\mathcal{R}(e)$  and  $\mathcal{R}'(e)$  are defined as follows:

- For all  $e' \in \mathcal{T}_q(S, \mathcal{I})$ ,  $\mathcal{R}(e)(e') = \mathcal{K}(\Sigma)(e, e')$  and  $\mathcal{R}'(e, e') = \mathcal{K}(\Sigma)(e', e)$ .
- For all  $f : e' \rightarrow e'', g : e \rightarrow e', h : e'' \rightarrow e \in Arr_{\Sigma}$ ,

$$f^{\mathcal{R}(e)}([g]_{\sim_{\Sigma}}) = [f \circ g]_{\sim_{\Sigma}} \quad \text{and} \quad f^{\mathcal{R}'(e)}([h]_{\sim_{\Sigma}}) = [h \circ f]_{\sim_{\Sigma}}.$$

Hence for all  $f : e \rightarrow e' \in Arr_{\Sigma}$ , a  $\Sigma$ -homomorphism  $h : \mathcal{R}(e) \rightarrow \mathcal{A}$  maps  $[f]_{\sim_{\Sigma}}$  to an element of  $\mathcal{A}(e')$ .

By (3),  $h$  uniquely represents an element of  $\mathcal{A}(e)$  and we have isos

$$\Phi : Alg_{\Sigma}(\mathcal{R}(e), \mathcal{A}) \rightarrow \mathcal{A}(e) \quad \text{and} \quad \Psi : \mathcal{A}(e) \rightarrow Alg_{\Sigma}(\mathcal{R}(e), \mathcal{A})$$

that are defined as follows (see the **Yoneda Lemma**):

For all  $\Sigma$ -homomorphisms  $h : \mathcal{R}(e) \rightarrow \mathcal{A}$ ,  $a \in \mathcal{A}(e)$  and  $f : e \rightarrow e' \in \text{Arr}_\Sigma$ ,

$$\Phi(h) = h_e([id_e]_{\sim_\Sigma}) \quad \text{and} \quad \Psi(a)_{e'}([f]_{\sim_\Sigma}) = f^{\mathcal{A}}(a). \quad (4)$$

Martin Brandenburg [35] gives a sociological interpretation of (3): Each “individual”  $a \in \mathcal{A}(e)$  is determined by its “contacts to the environment”, given by  $\Psi(a)$ , which, by (4), covers  $f^{\mathcal{A}}(a)$  for all  $\Sigma$ -arrows  $f$  with source  $e$ .

Analogously, a  $\Sigma$ -homomorphism  $h : \mathcal{R}'(e) \rightarrow \mathcal{A}$  is an  $S$ -sorted function maps (the  $\sim_\Sigma$ -equivalence class of) an arrow  $f : e' \rightarrow e$  to an element of  $\mathcal{A}(e')$ . Again, (3) implies that  $h$  uniquely represents an element of  $\mathcal{A}(e)$ .

## Lemma EMH

Let  $\mathcal{A}, \mathcal{B}, \mathcal{C}$  be  $\Sigma$ -algebras with carriers  $A, B, C$ , respectively. and  $g : A \rightarrow B$  and  $h : B \rightarrow C$  be  $S$ -sorted functions such that  $h \circ g$  is  $\Sigma$ -homomorphic.

- (1) If  $g$  is epi in  $\text{Alg}_\Sigma$ , then  $h$  is  $\Sigma$ -homomorphic.
- (2) If  $h$  is mono in  $\text{Alg}_\Sigma$ , then  $g$  is  $\Sigma$ -homomorphic.

*Proof.* (1) and (2) can be shown by diagram chasing. □

## Reducts

Let  $\sigma : \Sigma \rightarrow \Sigma'$  be a signature morphism and  $h : \mathcal{A} \rightarrow \mathcal{B}$  be a  $\Sigma'$ -homomorphism.

The  **$\sigma$ -reduct of  $\mathcal{A}$** ,  $\mathcal{A}|_\sigma$ , is the  $\Sigma$ -algebra that is defined as follows:

- For all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $\mathcal{A}|_\sigma(e) = \mathcal{A}(\sigma(e))$ .
- For all  $f \in F$ ,  $f^{\mathcal{A}|_\sigma} = \sigma(f)^{\mathcal{A}}$ .

The  **$\sigma$ -reduct of  $h$** ,  $h|_\sigma : \mathcal{A}|_\sigma \rightarrow \mathcal{B}|_\sigma$ , is the  $\Sigma$ -homomorphism that is defined as follows:

For all  $s \in S$ ,  $(h|_\sigma)_s = h_{\sigma(s)}$ .

$\sigma$ -reducts are the images of the **reduct functor**  $\underline{|}_\sigma : \text{Alg}_{\Sigma'} \rightarrow \text{Alg}_\Sigma$ .

If  $\sigma$  is an inclusion of signatures,  $\mathcal{A}|_\sigma$  and  $h|_\sigma$  are called  **$\Sigma$ -reducts** and written as  $\mathcal{A}|_\Sigma$  and  $h|_\Sigma$ , respectively. In this case,  $\underline{|}_\sigma$  coincides with the forgetful functor  $U_\Sigma : \text{Alg}_{\Sigma'} \rightarrow \text{Alg}_\Sigma$ .

## Sample algebras

The following algebras are supposed to interpret sum types by disjoint unions and product types by Cartesian products (see chapter 2).

1.  $\mathbb{N}$  is the carrier of the synonymous  $\text{Nat}$ -algebra  $\mathbb{N}$  whose operations

$$\text{zero}^{\mathbb{N}} : 1 \rightarrow \mathbb{N}, \quad \text{succ}^{\mathbb{N}} : \mathbb{N} \rightarrow \mathbb{N}$$

are defined as follows: For all  $n \in \mathbb{N}$ ,

$$\begin{aligned}\text{zero}^{\mathbb{N}}() &= 0, \\ \text{succ}^{\mathbb{N}}(n) &= n + 1.\end{aligned}$$

$\mathbb{N}$  is also the carrier of the  $\text{List}(X)$ -algebra  $\text{Length}$  whose operations

$$\alpha^{\text{Length}} : 1 \rightarrow \mathbb{N}, \quad \text{cons}^{\text{Length}} : X \times \mathbb{N} \rightarrow \mathbb{N}$$

are defined as follows: For all  $x \in X$  and  $n \in \mathbb{N}$ ,

$$\begin{aligned}\alpha^{\text{Length}}(x) &= 0, \\ \text{cons}^{\text{Length}}(x, n) &= n + 1.\end{aligned}$$

2.  $\mathbb{N}_\infty =_{def} \mathbb{N} \cup \{\omega\}$  and  $1^\infty = 1^* \cup 1^\mathbb{N}$  are carriers of the synonymous  $coNat$ -algebras whose operation

$$pred^{\mathbb{N}_\infty} : \mathbb{N}_\infty \rightarrow 1 + \mathbb{N}_\infty \quad \text{resp.} \quad pred^{1^\infty} : 1^\infty \rightarrow 1 + 1^\infty$$

are defined as follows: For all  $n > 0$ ,

$$\begin{array}{lll} pred^{\mathbb{N}_\infty}(0) & = () , & pred^{1^\infty}(\epsilon) & = () , \\ pred^{\mathbb{N}_\infty}(n) & = n - 1 , & pred^{1^\infty}((())^n) & = ()^{n-1} , \\ pred^{\mathbb{N}_\infty}(\infty) & = \infty , & pred^{1^\infty}(\lambda n.()) & = \lambda n.(). \end{array}$$

3. The set  $X^* \times Y$  of **guarded sequences** is the carrier of the  $Dyn(X, Y)$ -algebra  $Seq(X, Y)$  whose operations

$$cons^{Seq(X, Y)} : X \times (X^* \times Y) \rightarrow X^* \times Y, \quad \alpha^{Seq(X, Y)} : Y \rightarrow X^* \times Y$$

are defined as follows: For all  $w \in X^*$ ,  $x \in X$  and  $y \in Y$ ,

$$\begin{aligned} cons^{Seq(X, Y)}(x, (w, y)) &= (xw, y), \\ \alpha^{Seq(X, Y)}(y) &= (\epsilon, y). \end{aligned}$$

See [67] for the use of  $Seq(X, Y)$  for *functional* modelling in ecology and environmental science.

$X^*$  is the carrier of the synonymous  $List(X)$ -Algebra  $X^*$  whose operations

$$cons^{X^*} : X \times X^* \rightarrow X^*, \quad \alpha^{X^*} : 1 \rightarrow X^*$$

are defined as follows: For all  $x \in X$  and  $w \in X^*$ ,

$$\begin{aligned} cons^{X^*}(x, w) &= xw, \\ \alpha^{X^*}() &= \epsilon. \end{aligned}$$

$X^*$  is also the carrier of the  $Mon$ -algebra  $Word(X)$  whose operations

$$one^{Word(X)} : 1 \rightarrow X^*, \quad mul^{Word(X)} : X^* \times X^* \rightarrow X^*$$

are defined as follows: For all  $v, w \in X^*$ ,

$$\begin{aligned} one^{Word(X)}() &= \epsilon, \\ mul^{Word(X)}(v, w) &= vw. \end{aligned}$$

4.  $X^X$  is the carrier of the  $\text{Mon}$ -algebra  $\text{Endo}(X)$  whose operations

$$\text{one}^{\text{Endo}(X)} : 1 \rightarrow X^X, \text{ mul}^{\text{Endo}(X)} : X^X \times X^X \rightarrow X^X$$

are defined follows: For all  $f, g : X \rightarrow X$ ,

$$\begin{aligned}\text{one}^{\text{Endo}(X)}() &= \text{id}_X, \\ \text{mul}^{\text{Endo}(X)}(f, g) &= g \circ f.\end{aligned}$$

$\text{Word}(X)$  and  $\text{Endo}(X)$  are monoids.

5.  $X^{\mathbb{N}}$  is the carrier of the  $\text{Stream}(X)$ -algebra  $\text{InfSeq}(X)$  whose operations

$$\text{head}^{\text{InfSeq}(X)} : X^{\mathbb{N}} \rightarrow X, \text{ tail}^{\text{InfSeq}(X)} : X^{\mathbb{N}} \rightarrow X^{\mathbb{N}}$$

are defined as follows: For all  $f : \mathbb{N} \rightarrow X$ ,

$$\begin{aligned}\text{head}^{\text{InfSeq}(X)}(f) &= f(0), \\ \text{tail}^{\text{InfSeq}(X)}(f) &= \lambda n. f(n + 1).\end{aligned}$$

$X^{\mathbb{N}}$  is also the carrier of the  $coStream(X)$ -algebra  $coInfSeq(X)$  whose operation

$$\delta^{coInfSeq(X)} : X \times X^{\mathbb{N}} \rightarrow X^{\mathbb{N}}$$

is defined as follows: For all  $x \in X$ ,  $f : \mathbb{N} \rightarrow X$  and  $n > 0$ ,

$$\begin{aligned}\delta^{coInfSeq(X)}(x, f)(0) &= x, \\ \delta^{coInfSeq(X)}(x, f)(n) &= f(n - 1).\end{aligned}$$

6. The following  $Stream(\mathbb{Z})$ -algebra  $zo$  represents the periodic streams  $0, 1, 0, 1, \dots$  and  $1, 0, 1, 0, \dots$ :

$$\begin{aligned}zo_{list} &= \{Blink, Blink'\}, \\ head^{zo}(Blink) &= 0, \\ tail^{zo}(Blink) &= Blink', \\ head^{zo}(Blink') &= 1, \\ tail^{zo}(Blink') &= Blink.\end{aligned}$$

7. The following  $DAut(\mathbb{Z}, 2)$ -algebra  $eo$  is the minimal automaton that accepts a word  $x_1 \dots x_n$  over  $\mathbb{Z}$  in  $esum$  or  $osum$  iff  $\sum_{i=1}^n x_i$  is even or odd, respectively (see example EOSUM):

$$\begin{aligned}
 eo_{state} &= \{esum, osum\}, \\
 \delta^{eo}(esum) &= \lambda x. \text{if even}(x) \text{ then } esum \text{ else } osum, \\
 \delta^{eo}(osum) &= \lambda x. \text{if odd}(x) \text{ then } esum \text{ else } osum, \\
 \beta^{eo} &= \lambda st. \text{if } st = esum \text{ then } 1 \text{ else } 0.
 \end{aligned}$$

8.  $T = X^* \times Y \cup X^{\mathbb{N}}$  is the carrier of the  $coDyn(X, Y)$ -algebra  $coSeq(X, Y)$  whose operation

$$split^{coSeq(X, Y)} : T \rightarrow X \times T + Y$$

is defined as follows: For all  $x \in X$ ,  $w \in X^*$ ,  $y \in Y$  and  $f \in X^{\mathbb{N}}$ ,

$$\begin{aligned}
 split^{coSeq(X, Y)}(\epsilon, y) &= \iota_2(y), \\
 split^{coSeq(X, Y)}(xw, y) &= \iota_1(x, (w, y)), \\
 split^{coSeq(X, Y)}(f) &= \iota_1(f(0), \lambda n. f(n + 1)).
 \end{aligned}$$

See [67] for the use of  $Seq(X, Y)$  for *interactive* modelling in ecology and environmental science.

9.  $T = X^+ \cup X^{\mathbb{N}}$  is the carrier of the  $coNelist(X)$ -algebra  $Neseq(X)$  whose operation

$$split^{Neseq(X)} : T \rightarrow X \times T + 1$$

is defined as follows: For all  $x \in X$ ,  $w \in X^+$  and  $f \in X^{\mathbb{N}}$ ,

$$\begin{aligned} split^{Neseq(X)}(x) &= \iota_2(), \\ split^{Neseq(X)}(xw) &= \iota_1(x, w), \\ split^{Neseq(X)}(f) &= \iota_1(f(0), \lambda n. f(n + 1)). \end{aligned}$$

10.  $T = ftr(2, X)$  (see chapter 2) is the carrier of the  $Bintree(X)$ -algebra  $FBin(X)$  whose operations

$$bjoin^{FBin(X)} : X \times T \times T \rightarrow T, \quad empty^{FBin(X)} : 1 \rightarrow T$$

are defined as follows: For all  $f, g \in ftr(2, X)$ ,  $x \in X$  and  $w \in 2^*$ ,

$$\begin{aligned} bjoin^{FBin(X)}(x, f, g)(\epsilon) &= x, \\ bjoin^{FBin(X)}(x, f, g)(0w) &= f(w), \\ bjoin^{FBin(X)}(x, f, g)(1w) &= g(w), \\ empty^{FBin(X)} &= \Omega. \end{aligned}$$

11.  $T = X^{2^*}$  is the carrier of the  $\text{infBintree}(X)$ -algebra  $\text{InfBin}(X)$  whose operations

$$\text{left}, \text{right} : T \rightarrow T, \quad \text{root}^{\text{InfBin}(X)} : T \rightarrow X$$

are defined as follows: For all  $t \in T$ ,

$$\begin{aligned}\text{left}^{\text{InfBin}(X)}(t) &= \lambda w. t(0w), \\ \text{right}^{\text{InfBin}(X)}(t) &= \lambda w. t(1w), \\ \text{root}^{\text{InfBin}(X)}(t) &= t(\epsilon).\end{aligned}$$

12.  $T = \text{ltr}(2, X)$  (see chapter 2) is the carrier of the  $\text{coBintree}(X)$ -algebra  $\text{Bin}(X)$  whose operation

$$\text{split}^{\text{Bin}(X)} : T \rightarrow T \times X \times T + 1$$

is defined as follows: For all  $x \in X$  and  $t, u \in T$ ,

$$\begin{aligned}\text{split}^{\text{Bin}(X)}(\Omega) &= (), \\ \text{split}^{\text{Bin}(X)}(x\{0 \rightarrow t, 1 \rightarrow u\}) &= (x, t, u).\end{aligned}$$

13. Let  $T = otr(\mathbb{N}, <, X) \cap ftr(\mathbb{N}, X)$ .  $(T, T^*)$  is the carrier of the  $\text{Tree}(X)$ -algebra  $FTree(X)$  whose operations

$$join^{FTree(X)} : X \times T^* \rightarrow T, \ cons^{FTree(X)} : T \times T^* \rightarrow T^*, \ nil^{FTree(X)} : 1 \rightarrow T^*$$

are defined as follows: For all  $x \in X$ ,  $n > 0$ ,  $t, t_1, \dots, t_n \in T$ ,

$$\begin{aligned} join^{FTree(X)}(x, \epsilon) &= x, \\ join^{FTree(X)}(x, (t_1, \dots, t_n)) &= x(t_1, \dots, t_n), \\ cons^{FTree(X)}(t, \epsilon) &= t, \\ cons^{FTree(X)}(t, (t_1, \dots, t_n)) &= (t, t_1, \dots, t_n), \\ nil^{FTree(X)}() &= \epsilon. \end{aligned}$$

14.  $T = otr(\mathbb{N}, <, X) \cap wtr(\mathbb{N}, X)$  is the carrier of the  $\text{Tree}_\omega(X)$ -algebra  $WFTree(X)$  whose operation

$$join^{WFTree(X)} : X \times T^\infty \rightarrow T$$

is defined as follows:

For all  $x \in X$ ,  $n > 0$ ,  $t_1, \dots, t_n \in T$  and  $f \in T^{\mathbb{N}}$ ,

$$\begin{aligned} \text{join}^{WFTree(X)}(x, \epsilon) &= x, \\ \text{join}^{WFTree(X)}(x, (t_1, \dots, t_n)) &= x(t_1, \dots, t_n), \\ \text{join}^{WFTree(X)}(x, f) &= x\{n \rightarrow f(n) \mid n \in \mathbb{N}\}. \end{aligned}$$

15.  $T = ftr(Y, X)$  is the carrier of the  $ETree(X)$ -algebra  $FETree(X)$  whose operation

$$\text{join}^{FETree(X)} : X \times (Y \times T)^* \rightarrow T$$

is defined as follows: For all  $x \in X$ ,  $n > 0$  and  $y_1, \dots, y_n \in Y$  and  $t_1, \dots, t_n \in T$ ,

$$\begin{aligned} \text{join}^{FETree(X)}(x, \epsilon) &= x, \\ \text{join}^{FETree(X)}(x, ((y_1, t_1), \dots, (y_n, t_n))) &= x\{y_1 \rightarrow t_1, \dots, y_n \rightarrow t_n\}. \end{aligned}$$

16.  $T = otr(\mathbb{N}, <, X) \cap fbtr(\mathbb{N}, X) \cap itr(\mathbb{N}, X)$  is the carrier of the  $infTree(X)$ -algebra  $FBInfTree(X)$  whose operations

$$\begin{aligned} \text{subtrees}^{FBInfTree(X)} &: T \rightarrow T^+, \\ \text{root}^{FBInfTree(X)} &: T \rightarrow X \end{aligned}$$

are defined as follows: For all  $x \in X$ ,  $n > 0$  and  $t_1, \dots, t_n \in T$ ,

$$\begin{aligned} \text{subtrees}^{FBInfTree(X)}(x(t_1, \dots, t_n)) &= (t_1, \dots, t_n), \\ \text{root}^{FBInfTree(X)}(x(t_1, \dots, t_n)) &= x. \end{aligned}$$

17.  $T = otr(\mathbb{N}, <, X) \cap fbtr(\mathbb{N}, X)$  is the carrier of the  $coTree_\omega(X)$ -algebra  $FBTree(X)$  whose operations

$$subtrees^{FBTree(X)} : T \rightarrow T^*,$$

$$root^{FBTree(X)} : T \rightarrow X$$

are defined as follows: For all  $x \in X$ ,  $n > 0$  and  $t_1, \dots, t_n \in T$ ,

$$subtrees^{FBTree(X)}(x) = \epsilon,$$

$$subtrees^{FBTree(X)}(x(t_1, \dots, t_n)) = (t_1, \dots, t_n),$$

$$root^{FBTree(X)}(x) = x,$$

$$root^{FBTree(X)}(x(t_1, \dots, t_n)) = x.$$

18. Let  $T = otr(\mathbb{N}, <, X)$ .  $(T, T^\infty)$  is the carrier of the  $coTree(X)$ -algebra  $Tree_\infty(X)$  whose operations

$$\begin{aligned} subtrees^{Tree_\infty(X)} &: T \rightarrow T^\infty, \\ root^{Tree_\infty(X)} &: T \rightarrow X, \\ split^{Tree_\infty(X)} &: T^\infty \rightarrow T \times T^\infty + 1 \end{aligned}$$

are defined as follows:

For all  $x \in X$ ,  $n > 0$ ,  $t, t_1, \dots, t_n \in T$ ,  $w \in T^*$

and  $f \in T^\mathbb{N}$ ,

$$\begin{aligned} subtrees^{Tree_\infty(X)}(x) &= \epsilon, \\ subtrees^{Tree_\infty(X)}(x(t_1, \dots, t_n)) &= (t_1, \dots, t_n), \\ root^{Tree_\infty(X)}(x) &= x, \\ root^{Tree_\infty(X)}(x(t_1, \dots, t_n)) &= x, \\ split^{Tree_\infty(X)}(\epsilon) &= (), \\ split^{Tree_\infty(X)}(t \cdot w) &= (t, w), \\ split^{Tree_\infty(X)}(f) &= (f(0), \lambda n. f(n+1)). \end{aligned}$$

19. Let  $\Sigma = \text{Reg}(X)$ . The set  $\mathcal{P}(X^*)$  of **languages** is the carrier of the  $\text{Reg}(X)$ -algebra  $\text{Lang}(X)$  whose operations

$$\begin{aligned} \text{par}^{\text{Lang}(X)}, \text{seq}^{\text{Lang}(X)} &: \mathcal{P}(X^*) \times \mathcal{P}(X^*) \rightarrow \mathcal{P}(X^*), \\ \text{iter}^{\text{Lang}(X)} &: \mathcal{P}(X^*) \rightarrow \mathcal{P}(X^*), \\ \text{eps}^{\text{Lang}(X)} &: 1 \rightarrow \mathcal{P}(X^*), \\ \text{base}^{\text{Lang}(X)} &: \mathcal{P}(X) \rightarrow \mathcal{P}(X^*) \end{aligned}$$

are defined as follows:

For all  $L, L' \subseteq X^*$ ,  $b \in X$  and  $B \subseteq X$ ,

$$\begin{aligned} \text{par}^{\text{Lang}(X)}(L, L') &= L \cup L', \\ \text{seq}^{\text{Lang}(X)}(L, L') &= L \cdot L', \\ \text{iter}^{\text{Lang}(X)}(L) &= L^*, \\ \text{eps}^{\text{Lang}(X)} &= \{\epsilon\}, \\ \text{base}^{\text{Lang}(X)}(B) &= B. \end{aligned}$$

The usual semantics of a regular expression, i.e., a ground  $\text{Reg}(X)$ -term, say  $t$ , is obtained by folding  $t$  in  $\text{Lang}(X)$ .

20.  $\mathcal{P}(X^*)$  is also the carrier of the  $Acc(X)$ -algebra  $Pow(X)$  whose operations

$$\begin{aligned}\delta^{Pow(X)} &: \mathcal{P}(X^*) \rightarrow \mathcal{P}(X^*)^X, \\ \beta^{Pow(X)} &: \mathcal{P}(X^*) \rightarrow 2\end{aligned}$$

are defined as follows: For all  $L \subseteq X^*$  and  $x \in X$ ,

$$\begin{aligned}\delta^{Pow(X)}(L)(x) &= \{w \in X^* \mid x \cdot w \in L\}, \\ \beta^{Pow(X)}(L) &= \begin{cases} 1 & \text{if } \epsilon \in L, \\ 0 & \text{otherwise.} \end{cases}\end{aligned}$$

21.  $\mathcal{P}(X^*)$  is also the carrier of the  $NAcc(X)$ -algebra  $NPow(X)$  whose operations

$$\begin{aligned}\delta^{NPow(X)} &: \mathcal{P}(X^*) \rightarrow \mathcal{P}_\omega(\mathcal{P}(X^*))^X, \\ \beta^{NPow(X)} &: \mathcal{P}(X^*) \rightarrow 2\end{aligned}$$

are defined as follows: For all  $L \subseteq X^*$  and  $x \in X$ ,

$$\begin{aligned}\delta^{NPow(X)}(L)(x) &= \{\{w \in X^* \mid x \cdot w \in L\}\}, \\ \beta^{NPow(X)}(L) &= \begin{cases} 1 & \text{if } \epsilon \in L, \\ 0 & \text{otherwise.} \end{cases}\end{aligned}$$

22. 2 is the carrier of the  $Reg(X)$ -algebra  $Bool$  whose operations

$$\begin{aligned} par^{Bool}, seq^{Lang(X)} &: 2 \times 2 \rightarrow 2, \\ iter^{Bool} &: 2 \rightarrow 2, \\ eps^{Bool} &: 1 \rightarrow 2, \\ base^{Bool} &: \mathcal{P}(X) \rightarrow 2 \end{aligned}$$

are defined as follows: For all  $x, y \in 2$ ,  $b \in X$  and  $B \subseteq X$ ,

$$\begin{aligned} par^{Bool}(x, y) &= max\{x, y\}, \\ seq^{Bool}(x, y) &= x * y, \\ iter^{Bool}(x) &= 1, \\ eps^{Bool} &= 1, \\ base^{Bool}(B) &= 0. \end{aligned}$$

23.  $T_{Reg(X)}$  is the carrier of both the  $Reg(X)$ -algebra of ground  $Reg(X)$ -terms (“regular expressions”; see [Sample terms and coterms](#)) and the **Brzozowski automaton**  $Bro(X)$  for accepting regular languages [36, 87], i.e., the  $Acc(X)$ -algebra whose operations

$$\delta = \delta^{Bro(X)} : T_{Reg(X)} \rightarrow T_{Reg(X)}^X \quad \text{and} \quad \beta = \beta^{Bro(X)} : T_{Reg(X)} \rightarrow 2$$

are defined as follows:

For all  $t, u \in T_{Reg(X)}$ ,  $b \in X$  and  $B \subseteq X$ ,

$$\begin{aligned}
\delta(par(t, u)) &= \lambda x. par(\delta(t)(x), \delta(u)(x)), \\
\delta(seq(t, u)) &= \lambda x. if \beta(t) then par(seq(\delta(t)(x), u), \delta(u)(x)) \\
&\quad else seq(\delta(t)(x), u), \\
\delta(iter(t)) &= \lambda x. seq(\delta(t)(x), iter(t)), \\
\delta(eps) &= \lambda x. base(\emptyset), \\
\delta(base(B)) &= \lambda x. if x \in B then eps else base(\emptyset), \\
\beta(par(t, u)) &= max(\beta(t), \beta(u)), \\
\beta(seq(t, u)) &= \beta(t) * \beta(u), \\
\beta(iter(t)) &= 1, \\
\beta(eps) &= 1, \\
\beta(base(B)) &= 0.
\end{aligned}$$

$\delta(t)$  and  $\beta(t)$  are called the *Brzozowski derivative* [36, 87] and *initial value* of  $t$ , respectively. For a proof that these equations define  $\delta^{Bro(X)}$  and  $\beta^{Bro(X)}$  uniquely on  $T_{Reg(X)}$ , see [Sample inductively defined functions](#).

24. The set  $Y^{X^*}$  of **behavior functions** is the carrier of the  $DAut(X, Y)$ -algebra  $Beh(X, Y)$  whose operations

$$\begin{aligned}\delta^{Beh(X,Y)} &: Y^{X^*} \rightarrow (Y^{X^*})^X, \\ \beta^{Beh(X,Y)} &: Y^{X^*} \rightarrow Y\end{aligned}$$

are defined as follows: For all  $f : X^* \rightarrow Y$  and  $x \in X$ ,

$$\begin{aligned}\delta^{Beh(X,Y)}(f)(x) &= \lambda w. f(x \cdot w), \\ \beta^{Beh(X,Y)}(f) &= f(\epsilon).\end{aligned}$$

Hence the function  $set2fun : \mathcal{P}(X^*) \rightarrow 2^{X^*}$  that maps each language  $L$  over  $X$  to its indicator function is a  $Acc(X)$ -isomorphism from  $Pow(X)$  to  $Beh(X, 2)$  (see above).

$set2fun$  is also a  $Reg(X)$ -homomorphism from  $Lang(X)$  to  $Lang'(X)$  where  $Lang'(X)$  has the carrier  $2^{X^*}$  and interprets the operations of  $Reg(X)$  as follows:

For all  $f, g : X^* \rightarrow 2$ ,  $w \in X^*$ ,  $v \in X^+$ ,  $b \in B$  and  $B \subseteq X$ ,

$$par^{Lang'(X)}(f, g)(w) = \max\{f(w), g(w)\},$$

$$\begin{aligned}
 seq^{Lang'(X)}(f, g)(w) &= \max\left(\left\{\begin{array}{l} \{f(\epsilon) * g(w) \mid w \in X^+\} \cup \\ \{f(w) * g(\epsilon) \mid w \in X^+\} \cup \\ \{f(w_1) * g(w_2) \mid w_1, w_2 \in X^+, w_1 w_2 = w\} \end{array}\right\}), \\
 iter^{Lang'(X)}(f)(\epsilon) &= 1, \\
 iter^{Lang'(X)}(f)(v) &= \max\{f(w_1) * \dots * f(w_n) \mid w_1, \dots, w_n \in X^+, w_1 \dots w_n = v\}, \\
 eps^{Lang'(X)}(w) &= \text{if } w = \epsilon \text{ then } 1 \text{ else } 0, \\
 base^{Lang'(X)}(B)(w) &= \text{if } w \in B \text{ then } 1 \text{ else } 0.
 \end{aligned}$$

Haskell implementation: `regBeh`

**25.** A commutative monoid  $(M, +, 0)$  is a **semiring** if there are a constant  $1 \in M$  and a function  $* : M^2 \rightarrow M$  called *multiplication* such that  $*$  are associative,  $*$  distributes over  $+$ ,  $1$  is the identity w.r.t.  $*$  and for all  $m \in M$ ,  $0 * m = 0 = m * 0$  (multiplication with  $0$  *annihilates*  $M$ ). For instance,  $(2, \max, 0)$ ,  $(\mathbb{N}, +, 0)$  and  $(\mathbb{R}, +, 0)$  are semirings.

Let  $(R, +, 0, *, 1)$  be a semiring. A commutative monoid  $(A, +, 0)$  is an  **$R$ -semimodule** if there is a function  $\cdot : R \times A \rightarrow A$  called *scalar multiplication* or  **$R$ -action** such that  $\cdot$  distributes over  $+$  and for all  $r, s \in R$  and  $a \in A$ ,  $(r * s) \cdot a = r \cdot (s \cdot a)$ ,  $1 \cdot a = a$  and  $0 \cdot a = r \cdot 0$  (multiplication with 0 annihilates  $A$  or  $R$ ).

Every semiring  $R$  is an  $R$ -semimodule. (1)

Given a set  $X$ ,  $R^X$  and  $R_\omega^X$  are also  **$R$ -semimodules** where addition, zero and scalar multiplication are defined as follows: For all  $f, g : X \rightarrow_\omega R$ ,  $x \in X$  and  $r \in R$ ,  $(f + g)(x) = f(x) + g(x)$ ,  $0(x) = 0$  and  $(r \cdot f)(x) = r \cdot f(x)$ . (2)

Let  $(R, +, 0, *, 1)$  be a semiring.

A function  $h : A \rightarrow B$  between two  $R$ -semimodules  $A$  and  $B$  is **linear** (w.r.t.  $R$ ) if for all  $x, y \in A$  and  $r \in M$ ,

$$h(x + y) = h(x) + h(y) \quad \text{and} \quad h(r \cdot x) = r \cdot h(x).$$

**$SMod_R$**  denotes the category of  $R$ -semimodules and  $R$ -linear functions.

A  $DAut(X, R)$ -algebra  $\mathcal{A}$  is a **linear automaton** if the carrier of  $\mathcal{A}$  is an  $R$ -semimodule and  $\delta, \beta$  are linear.

Since  $R^X$  is an  $R$ -semimodule,  $R^{X^*}$  and  $(R^{X^*})^X$  are also  $R$ -semimodules. The functions of  $R^{X^*}$  are called **formal power series** (see [145]).

Moreover,  $\delta^{Beh(X,Y)}$  and  $\beta^{Beh(X,Y)}$  are linear and thus  $Beh(X, Y)$  is a linear automaton.

**26.** The set  $Y^{X^+}$  is the carrier of the *Mealy*( $X, Y$ )-algebra  $MBeh(X, Y)$  whose operations

$$\begin{aligned}\delta^{MBeh(X,Y)} &: Y^{X^+} \rightarrow (Y^{X^+})^X, \\ \beta^{MBeh(X,Y)} &: Y^{X^+} \rightarrow Y^X\end{aligned}$$

are defined as follows: For all  $f : X^+ \rightarrow Y$ ,  $x \in X$  and  $w \in X^+$ ,

$$\begin{aligned}\delta^{MBeh(X,Y)}(f)(x)(w) &= f(x \cdot w), \\ \beta^{MBeh(X,Y)}(f)(x) &= f(x).\end{aligned}$$

The set  $C(X, Y)$  of causal functions from  $X^{\mathbb{N}}$  to  $Y^{\mathbb{N}}$  (see chapter 2) is the carrier of the *Mealy*( $X, Y$ )-algebra  $Causal(X, Y)$  whose operations

$$\begin{aligned}\delta^{Causal(X,Y)} &: C(X, Y) \rightarrow C(X, Y)^X, \\ \beta^{Causal(X,Y)} &: C(X, Y) \rightarrow Y^X\end{aligned}$$

are defined as follows:

For all  $f \in \mathcal{C}(X, Y)$  and  $x \in X$ ,

$$\begin{aligned}\delta^{Causal(X,Y)}(f)(x) &= tail \circ \lambda s. f(x \cdot s), \\ \beta^{Causal(X,Y)}(f)(x) &= head \circ \lambda s. f(x \cdot s).\end{aligned}$$

$MBeh(X, Y)$  and  $Causal(X, Y)$  are  $Mealy(X, Y)$ -homomorphic.

**27.**  $ltr(X, Y)$  is the carrier of the  $PAut(X, Y)$ -algebra  $PBeh(X, Y)$  whose operations

$$\begin{aligned}\delta^{PBeh(X,Y)} &: ltr(X, Y) \rightarrow (1 + ltr(X, Y))^X, \\ \beta^{PBeh(X,Y)} &: ltr(X, Y) \rightarrow Y\end{aligned}$$

are defined as follows: For all  $Z \subseteq X$ ,  $t = y\{x \rightarrow t_x \mid x \in Z\} \in ltr(X, Y)$  and  $x \in X$ ,

$$\begin{aligned}\delta^{PBeh(X,Y)}(t)(x) &= \begin{cases} t_x & \text{if } x \in Z, \\ () & \text{otherwise,} \end{cases} \\ \beta^{PBeh(X,Y)}(t) &= y.\end{aligned}$$

**28.**  $T = otr(X \times \mathbb{N}, (\Delta_X, <), Y)$  is the carrier of the  $NAut^*(X, Y)$ -algebra  $NBeh(X, Y)$  whose operations

$$\delta^{NBeh(X,Y)} : T \rightarrow (T^*)^X, \quad \beta^{NBeh(X,Y)} : T \rightarrow Y$$

are defined as follows:

For all  $Z \subseteq X$  and  $n \in \mathbb{N}$ ,  $t = y\{(x, i) \rightarrow t_{x,i} \mid x \in Z, 1 \leq i \leq n\} \in T$  and  $x \in X$ ,

$$\begin{aligned}\delta^{NBeh(X,Y)}(t)(x) &= \begin{cases} (t_{x,1}, \dots, t_{x,n}) & \text{if } x \in Z, \\ \epsilon & \text{otherwise,} \end{cases} \\ \beta^{NBeh(X,Y)}(t) &= y.\end{aligned}$$

Let  $\Sigma = (S, \mathcal{I}, C)$  be a finitary signature. For all  $c : s_1 \times \dots \times s_n \rightarrow s \in C$ ,  $S$ -sorted subsets  $L$  of  $T_{\Sigma,s}$  (see  $\Sigma$ -terms and -coterm) and  $1 \leq i \leq n$ ,

$$sucs(c, L)_i =_{def} \{t_i \in T_{\Sigma, s_i} \mid \forall j \in [n] \setminus \{i\} \exists t_j \in T_{\Sigma, s_j} : c(t_1, \dots, t_n) \in L\}.$$

**29.**  $\mathcal{P}(T_\Sigma) =_{def} (\mathcal{P}(T_{\Sigma,s}))_{s \in S \cup \mathcal{I}}$  is the carrier of the  $TAcc(\Sigma)$ -algebra  $TPow(\Sigma)$  whose operations

$$\delta_c^{TPow(\Sigma)} : \mathcal{P}(T_{\Sigma,s}) \rightarrow \mathcal{P}(T_{\Sigma, s_1}) \times \dots \times \mathcal{P}(T_{\Sigma, s_n}), \quad c : s_1 \times \dots \times s_n \rightarrow s \in C,$$

are defined as follows: For all  $L \subseteq T_{\Sigma,s}$ ,  $\delta_c^{TPow(\Sigma)}(L) = (sucs(c, L)_1, \dots, suc(c, L)_n)$ .

**30.**  $\mathcal{P}(T_\Sigma)$  is also the carrier of the  $NTAcc(\Sigma)$ -algebra  $NTPow(\Sigma)$  whose operations

$$\delta_c^{NTPow(\Sigma)} : \mathcal{P}(T_{\Sigma,s}) \rightarrow \mathcal{P}_\omega(\mathcal{P}(T_{\Sigma, s_1}) \times \dots \times \mathcal{P}(T_{\Sigma, s_n})), \quad c : s_1 \times \dots \times s_n \rightarrow s \in C,$$

are defined as follows: For all  $L \subseteq T_{\Sigma,s}$ ,  $\delta_c^{NTPow(\Sigma)}(L) = \{(sucs(c, L)_1, \dots, suc(c, L)_n)\}$ .

31.  $\mathcal{P}(T_\Sigma)$  is also the carrier of the  $NTAcc^*(\Sigma)$ -algebra  $NTPow^*(\Sigma)$  whose operations

$$\delta_c^{NTPow(\Sigma)} : \mathcal{P}(T_{\Sigma,s}) \rightarrow (\mathcal{P}(T_{\Sigma,s_1}) \times \dots \times \mathcal{P}(T_{\Sigma,s_n}))^*, \quad c : s_1 \times \dots \times s_n \rightarrow s \in C,$$

are defined as follows: For all  $L \subseteq T_{\Sigma,s}$ ,  $\delta_c^{NTPow^*(\Sigma)}(L) = [(sucs(c, L)_1, \dots, suc(c, L)_n)]$ .

## Products, sums, invariants and quotients of $\Sigma$ -algebras

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature and  $\mathcal{A} = (\mathcal{A}_i)_{i \in I}$  be a tuple of  $\Sigma$ -algebras.

### Products

The product of  $\mathcal{A}$  in  $Alg_\Sigma$  is given by the  $\Sigma$ -algebra  $\mathcal{B} = \prod_{i \in I} \mathcal{A}_i$  that is defined as follows:

- For all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $\mathcal{B}(e) = \bigtimes_{i \in I} \mathcal{A}_i(e)$ .
- For all  $f : e \rightarrow e' \in F$ ,  $f^{\mathcal{B}} = \prod_{i \in I} f^{\mathcal{A}_i} : \mathcal{B}(e) \rightarrow \mathcal{B}(e')$ .

For all  $i \in I$ , the projection  $\pi_i =_{def} (\pi_{i,e} : \mathcal{B}(e) \rightarrow \mathcal{A}_i(e))_{e \in \mathcal{T}_q(S, \mathcal{I})}$  is  $\Sigma$ -homomorphic: For all  $f : e \rightarrow e' \in F$ ,

$$\pi_{i,e'} \circ f^{\mathcal{B}} = \pi_{i,e'} \circ \prod_{i \in I} f^{\mathcal{A}_i} \stackrel{(7) \text{ in chapter 2}}{=} f^{\mathcal{A}_i} \circ \pi_{i,e}.$$

Let  $\mathcal{C}$  be a  $\Sigma$ -algebra and  $(h_i : \mathcal{C} \rightarrow \mathcal{A}_i)_{i \in I}$  be a tuple of  $\Sigma$ -homomorphisms. Then  $\langle h_i \rangle_{i \in I} =_{def} (\langle h_{i,e} \rangle_{i \in I} : \mathcal{C}(e) \rightarrow \mathcal{B}(e))_{e \in \mathcal{T}_q(S, \mathcal{I})}$  is also  $\Sigma$ -homomorphic:

For all  $i \in I$  and  $f : e \rightarrow e' \in F$ ,

$$(\langle h_i \rangle_{i \in I})_{e'} \circ f^{\mathcal{C}} = \langle h_{i,e'} \rangle_{i \in I} \circ f^{\mathcal{C}} \stackrel{(6) \text{ in chapter 2}}{=} \langle h_{i,e'} \circ f^{\mathcal{C}} \rangle_{i \in I} \stackrel{h_i \text{ is } \Sigma\text{-hom.}}{=} \langle f^{\mathcal{A}_i} \circ h_{i,e} \rangle_{i \in I}$$

$$\begin{aligned}
 &= \langle f^{\mathcal{A}_i} \circ \pi_{i,e} \circ \langle h_{i,e} \rangle_{i \in I} \rangle_{i \in I} \stackrel{(6) \text{ in chapter 2}}{=} \langle f^{\mathcal{A}_i} \circ \pi_{i,e} \rangle_{i \in I} \circ \langle h_{i,e} \rangle_{i \in I} = \prod_{i \in I} f^{\mathcal{A}_i} \circ \langle h_{i,e} \rangle_{i \in I} \\
 &\stackrel{\text{Def. } f^{\mathcal{B}}}{=} f^{\mathcal{B}} \circ \langle h_{i,e} \rangle_{i \in I} = f^{\mathcal{B}} \circ (\langle h_i \rangle_{i \in I})_e.
 \end{aligned}$$

Uniqueness of  $\langle h_i \rangle_{i \in I}$  follows from uniqueness of  $\langle h_i \rangle_{i \in I}$  as a  $\mathcal{T}_q(S, \mathcal{I})$ -sorted function.

### Example $\prod Dyn$

Let  $\mathcal{A}$  be a  $coStream(X)$ -algebra (see chapter 8) with carrier  $Q$  and  $Y$  be a set.

The  $coStream(X)$ -algebra  $\mathcal{B} = \mathcal{A}^{(Q^Y)}$  and its extension to a  $Dyn(X, Y)$ -algebra are defined as follows:

- $\mathcal{B}(state) = Q^{(Q^Y)}$ .
- For all  $f : Y \rightarrow Q$ ,  $g : Q^Y \rightarrow Q$  and  $x \in X$ ,

$$\begin{aligned}
 cons^{\mathcal{B}}(x, g)(f) &= \pi_f(cons^{\mathcal{B}}(x, g)) = \pi_f(\langle cons^{\mathcal{A}} \circ \pi_{f, X \times state} \rangle_{f: Y \rightarrow Q}(x, g)) \\
 &= cons^{\mathcal{A}}(\pi_{f, X \times state}(x, g)) = cons^{\mathcal{A}}(x, \pi_f(g)) = cons^{\mathcal{A}}(g(f), x).
 \end{aligned}$$

- For all  $y \in Y$  and  $f : Y \rightarrow Q$ ,  $\alpha^{\mathcal{B}}(y)(f) = f(y)$ .

[20], Def. 4, provides this product automaton for the case  $Y = 1$ .

Given a semiring  $R$ , [149], section 3 defines a *weighted version* of  $\mathcal{B}$  for  $Y = 1$ :

Let  $\mathcal{A}$  be a  $WcoStream(X, R)$ -algebra (see chapter 8) with carrier  $Q$  and

$$(\delta^{\mathcal{A}})^*: X \times R_{\omega}^Q \rightarrow R_{\omega}^Q$$

be the  $SMod_R$ -extension of  $\delta^{\mathcal{A}}$  (see section 9.25). Then  $\mathcal{A}^*$  with  $\mathcal{A}^*(state) = R_{\omega}^Q$  and  $\delta^{\mathcal{A}^*} = (\delta^{\mathcal{A}})^*$  is a  $coStream(X)$ -algebra. Moreover, the product  $coStream(X)$ -algebra  $\mathcal{B} = (\mathcal{A}^*)^Q$  and its extension to a  $Dyn(X, 1)$ -algebra are defined as follows:

- $\mathcal{B}(state) = (R_{\omega}^Q)^Q$ .
- For all  $q \in Q$ ,  $g : Q \rightarrow R_{\omega}^Q$  and  $x \in X$ ,

$$\begin{aligned} cons^{\mathcal{B}}(x, g)(q) &= \pi_q(cons^{\mathcal{B}}(x, g)) = \pi_q(\langle cons^{\mathcal{A}^*} \circ \pi_{q, X \times state} \rangle_{q \in Q}(x, g)) \\ &= cons^{\mathcal{A}^*}(\pi_{q, X \times state}(x, g)) = \delta^{\mathcal{A}^*}(\pi_q(g), x) = \delta^{\mathcal{A}^*}(g(q), x). \end{aligned}$$

- For all  $q \in Q$ ,  $\alpha^{\mathcal{B}}(\epsilon)(q) = 1 \cdot q$  (see Preliminaries).

□

## Sums

The sum of  $\mathcal{A}$  in  $Alg_{\Sigma}$  is given by the  $\Sigma$ -algebra  $\mathcal{B} = \coprod_{i \in I} \mathcal{A}_i$  that is defined as follows (see, e.g., [143], section 4.1; [128], section 2.2.1; or [77], Proposition 2.1.5):

- For all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $\mathcal{B}(e) = \biguplus_{i \in I} \mathcal{A}_i(e)$ .

- For all  $f : e \rightarrow e' \in F$ ,  $f^{\mathcal{B}} = \coprod_{i \in I} f^{\mathcal{A}_i} : \mathcal{B}(e) \rightarrow \mathcal{B}(e')$ .

For all  $i \in I$ , the injection  $\iota_i = (\iota_{i,e} : \mathcal{A}_i(e) \rightarrow \mathcal{B}(e))_{e \in \mathcal{T}_q(S,\mathcal{I})}$  is  $\Sigma$ -homomorphic: For all  $f : e \rightarrow e' \in F$ ,

$$f^{\mathcal{B}} \circ \iota_{i,e} = \coprod_{i \in I} f^{\mathcal{A}_i} \circ \iota_{i,e} \stackrel{(18) \text{ in chapter 2}}{=} \iota_{i,e'} \circ f^{\mathcal{A}_i}.$$

Let  $\mathcal{C}$  be a  $\Sigma$ -algebra and  $(h_i : \mathcal{A}_i \rightarrow \mathcal{C})_{i \in I}$  be a tuple of  $\Sigma$ -homomorphisms. Then  $[h_i]_{i \in I} = ([h_{i,e}]_{i \in I} : \mathcal{C}(e) \rightarrow \mathcal{B}(e))_{e \in \mathcal{T}_q(S,\mathcal{I})}$  is also  $\Sigma$ -homomorphic:

For all  $i \in I$  and  $f : e \rightarrow e' \in F$ ,

$$\begin{aligned} ([h_i]_{i \in I})_{e'} \circ f^{\mathcal{B}} &= [h_{i,e'}]_{i \in I} \circ f^{\mathcal{B}} \stackrel{Def. f^{\mathcal{B}}}{=} [h_{i,e'}]_{i \in I} \circ \coprod_{i \in I} f^{\mathcal{A}_i} = [h_{i,e'}]_{i \in I} \circ [\iota_{i,e'} \circ f^{\mathcal{A}_i}]_{i \in I} \\ &\stackrel{(17) \text{ in chapter 2}}{=} [[h_{i,e'}]_{i \in I} \circ \iota_{i,e'} \circ f^{\mathcal{A}_i}]_{i \in I} = [h_{i,e'} \circ f^{\mathcal{A}_i}]_{i \in I} \stackrel{h_i \text{ is } \Sigma\text{-hom.}}{=} [f^{\mathcal{C}} \circ h_{i,e}]_{i \in I} \\ &\stackrel{(17) \text{ in chapter 2}}{=} f^{\mathcal{C}} \circ [h_{i,e}]_{i \in I} = f^{\mathcal{C}} \circ ([h_i]_{i \in I})_e. \end{aligned}$$

Uniqueness of  $[h_i]_{i \in I}$  follows from uniqueness of  $[h_i]_{i \in I}$  as a  $\mathcal{T}_q(S,\mathcal{I})$ -sorted function.

Example  $\coprod DAut$

Let  $\mathcal{A}$  be a  $Med(X)$ -algebra (see chapter 8) with carrier  $Q$  and  $Y$  be a set.

The  $Med(X)$ -algebra  $\mathcal{B} = \mathcal{A} \times Y^Q$  and its extension to a  $DAut(X, Y)$ -algebra are defined as follows:

- $\mathcal{B}(state) = Q \times Y^Q$ .
- For all  $q \in Q$ ,  $f : Q \rightarrow Y$  and  $x \in X$ ,

$$\begin{aligned}\delta^{\mathcal{B}}(q, f)(x) &= [\iota_{f, state}^X \circ \delta^{\mathcal{A}}]_{f: Q \rightarrow Y}(q, f)(x) = [\iota_{f, state}^X \circ \delta^{\mathcal{A}}]_{f: Q \rightarrow Y}(\iota_f(q))(x) \\ &= \iota_{f, state}^X(\delta^{\mathcal{A}}(q))(x) = \iota_{f, state}^X(\delta^{\mathcal{A}}(q))(x) = \iota_f(\delta^{\mathcal{A}}(q)(x)) = (\delta^{\mathcal{A}}(q)(x), f).\end{aligned}$$

- For all  $q \in Q$ , and  $f : Q \rightarrow Y$ ,  $\beta^{\mathcal{B}}(q, f) = f(q)$ .

[20], Def. 5, provides this sum automaton for the case  $Y = 2$ .

Given a semiring  $R$ , [149], section 3 defines a *weighted version* of  $\mathcal{B}$ :

Let  $\mathcal{A}$  be a  $WMed(X, R)$ -algebra (see chapter 8) with carrier  $Q$  and

$$(\delta^{\mathcal{A}})^* : R_{\omega}^Q \rightarrow (R_{\omega}^Q)^X$$

be the  $SMod_R$ -extension of  $\delta^{\mathcal{A}}$  (see section 9.25). Then  $\mathcal{A}^*$  with  $\mathcal{A}^*(state) = R_{\omega}^Q$  and  $\delta^{\mathcal{A}^*} = (\delta^{\mathcal{A}})^*$  is a  $Med(X)$ -algebra. Moreover, the sum  $Med(X)$ -algebra  $\mathcal{B} = \mathcal{A}^* \times R^Q$  and its extension to a  $DAut(X, R)$ -algebra are defined as follows:

- $\mathcal{B}(state) = R_{\omega}^Q \times R^Q$ .

- For all  $f : Q \rightarrow R$ ,  $g : Q \rightarrow_{\omega} R$  and  $x \in X$ ,

$$\begin{aligned}\delta^{\mathcal{B}}(g, f)(x) &= [\iota_{f, state^X} \circ \delta^{\mathcal{A}^*}]_{f: Q \rightarrow R}(q, g)(x) = [\iota_{f, state^X} \circ \delta^{\mathcal{A}^*}]_{f: Q \rightarrow R}(\iota_f(g))(x) \\ &= \iota_{f, state^X}(\delta^{\mathcal{A}^*}(g))(x) = \iota_{f, state}^X(\delta^{\mathcal{A}^*}(g))(x) = \iota_f(\delta^{\mathcal{A}^*}(g)(x)) = (\delta^{\mathcal{A}^*}(g)(x), f).\end{aligned}$$

- For all  $f : Q \rightarrow R$  and  $g : Q \rightarrow_{\omega} R$ ,  $\beta^{\mathcal{B}}(g, f) = f^*(g)$  where  $f^* : R_{\omega}^Q \rightarrow R$  is the  $SMod_R$ -extension of  $f$  (see section 9.25). □

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .

## Invariants

An  $S$ -sorted subset  $B$  of  $A$  is a  **$\Sigma$ -invariant** of  $\mathcal{A}$  if for all  $f : e \rightarrow e' \in F$  and  $a \in B_e$ ,  $f^{\mathcal{A}}(a) \in B_{e'}$ .

If  $\Sigma$  is constructive, then  $\lambda s.\emptyset$  is the least  $\Sigma$ -invariant of  $\mathcal{A}$  and  $\lambda s.A_s$  is the greatest  $\Sigma$ -invariant of  $\mathcal{A}$ .

Given  $a \in A$ , the least invariant of  $\mathcal{A}$  that contains  $a$  is denoted by  $\langle a \rangle$ .

For the construction of  $\langle a \rangle$  if  $\Sigma$  is destructive, see [State unfolding](#).

Every  $\Sigma$ -invariant  $B$  of  $\mathcal{A}$  induces the  **$\Sigma$ -subalgebra**  $\mathcal{A}|_B$  of  $\mathcal{A}$ :

- For all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $(\mathcal{A}|_B)(e) =_{def} B_e$ .
- For all  $f : e \rightarrow e' \in F$  and  $a \in B_e$ ,  $f^{\mathcal{A}|_B}(a) =_{def} f^{\mathcal{A}}(a)$ .

Moreover, the inclusion map  $inc : B \rightarrow A$  is a  $\Sigma$ -homomorphism from  $\mathcal{A}|_B$  to  $\mathcal{A}$ .

## Quotients

Let  $\mathcal{A}, \mathcal{B}$  be  $\Sigma$ -algebras with carriers  $A$  resp.  $B$ .

An  $S$ -sorted relation  $R \subseteq A \times B$  is a  **$\Sigma$ -bisimulation** if for all  $f : e \rightarrow e' \in F$  and  $(a, b) \in R_e$ ,  $(f^{\mathcal{A}}(a), f^{\mathcal{B}}(b)) \in R_{e'}$ . If  $\mathcal{A} = \mathcal{B}$ , then  $R$  is called a  $\Sigma$ -bisimulation **on**  $\mathcal{A}$ .

A  $\Sigma$ -bisimulation on  $\mathcal{A}$  is called a  **$\Sigma$ -congruence** if it is an equivalence relation.

If  $\Sigma$  is constructive, then  $\lambda s. \Delta_{A_s}^2$  is the least  $\Sigma$ -congruence on  $\mathcal{A}$  and  $\lambda s. A_s^2$  is the greatest  $\Sigma$ -congruence on  $\mathcal{A}$ .

A  $\Sigma$ -congruence  $R$  induces the  **$\Sigma$ -quotient (algebra)**  $\mathcal{A}/R$  of  $\mathcal{A}$  by  $R$ :

- For all  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $(\mathcal{A}/R)(e) =_{def} A_e/R_e$ .
- For all  $f : e \rightarrow e' \in F$  and  $a \in A_e$ ,  $f^{\mathcal{A}/R}([a]_R) =_{def} [f^{\mathcal{A}}(a)]_R$ .

Moreover, the natural map  $nat : A \rightarrow A/R$  that sends  $a \in A$  to  $[a]_R$  is a  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $\mathcal{A}/R$ . This implies

## Lemma CONGCL

Let  $R$  be a  $\Sigma$ -congruence. For all  $f : e \rightarrow e' \in Arr_\Sigma$  and  $(a, b) \in R$ ,  $(f^{\mathcal{A}}(a), f^{\mathcal{A}}(b)) \in R$ .

*Proof.*

$$\begin{aligned} nat(f^{\mathcal{A}}(a)) &\stackrel{Lemma}{=} f^{\mathcal{A}/R}(nat(a)) = f^{\mathcal{A}/R}([a]_R) = f^{\mathcal{A}/R}([b]_R) \\ &= f^{\mathcal{A}/R}(nat(b)) \stackrel{Lemma}{=} nat(f^{\mathcal{A}}(b)), \end{aligned}$$

i.e.,  $(f^{\mathcal{A}}(a), f^{\mathcal{A}}(b)) \in R$ .

□

## Theorem BISIM

- (1) Let  $\sim$  be a  $\Sigma$ -bisimulation on  $A$ . Then the equivalence closure  $\sim^{eq}$  of  $\sim$  (see chapter 3) is a  $\Sigma$ -congruence.
- (2) The *greatest*  $\Sigma$ -bisimulation on  $A$ , called  **$\Sigma$ -bisimilarity**, is an equivalence relation and thus agrees with the greatest  $\Sigma$ -congruence on  $A$ .

(3) A  $Mod(S, \mathcal{I})$ -morphism  $h : A \rightarrow B$  is a  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $\mathcal{B}$  iff the  $S$ -sorted relation

$$\text{graph}(h) =_{\text{def}} (\{(a, h_s(a)) \mid a \in A_s\})_{s \in S}$$

is a  $\Sigma$ -bisimulation.

*Proof.* Given an  $S$ -sorted binary relation  $\sim$  on  $A$ ,  $\sim^{eq}$  is the least fixpoint of

$$\begin{aligned} \Phi(\sim) : \mathsf{X}_{e \in \mathcal{T}_q(S, \mathcal{I})} \mathcal{P}(A_e^2) &\rightarrow \mathsf{X}_{e \in \mathcal{T}_q(S, \mathcal{I})} \mathcal{P}(A_e^2) \\ R &\mapsto (\sim_e \cup \Delta_{A_e}^2 \cup R_e^{-1} \cup R_e \cdot R_e)_{e \in \mathcal{T}_q(S, \mathcal{I})}. \end{aligned}$$

Moreover, let  $\text{sucs}(\sim)$  be the  $\mathcal{T}_q(S, \mathcal{I})$ -sorted set defined by

$$\text{sucs}(\sim)_e = \{(a, b) \in A_e^2 \mid \forall f : e \rightarrow e' \in F : f^A(a) \sim_{e'} f^A(b)\}$$

for all  $e \in \mathcal{T}_q(S, \mathcal{I})$ .

(1) Since  $\sim^{eq}$  is an equivalence relation, it remains to show that  $\sim^{eq}$  is a  $\Sigma$ -bisimulation, which holds true iff

$$\sim^{eq} \subseteq \text{sucs}(\sim^{eq}). \tag{3}$$

Since  $\sim^{eq} = \text{lfp}(\Phi(\sim))$ , fixpoint induction (see chapter 3) implies (3) if  $\text{sucs}(\sim)$  is  $\Phi(\sim)$ -closed, i.e., if  $\Phi(\sim)(\text{sucs}(\sim)) \subseteq \text{sucs}(\sim)$ .

So let  $(a, b) \in \Phi(\sim)(\text{sucs}(\sim^{eq}))$ . Hence  $(a, b) \in \text{sucs}(\sim^{eq})$  or we have one of the following three cases:

*Case 1:*  $a = b$ . Then for all  $f : e \rightarrow e' \in F$ ,  $f^A(a) = f^A(b)$  and thus  $f^A(a) \sim^{eq} f^A(b)$  because  $\sim^{eq}$  is reflexive.

*Case 2:*  $b \sim^{eq} a$ . Then for all  $f : e \rightarrow e' \in F$ ,  $f^A(b) \sim^{eq} f^A(a)$ . Hence  $f^A(a) \sim^{eq} f^A(b)$  because  $\sim^{eq}$  is symmetric.

*Case 3:*  $a \sim^{eq} c$  and  $c \sim^{eq} b$  for some  $c \in A$ . Then for all  $f : e \rightarrow e' \in F$ ,

$$f^A(a) \sim^{eq} f^A(c) \quad \text{and} \quad f^A(c) \sim^{eq} f^A(b).$$

Hence  $f^A(a) \sim^{eq} f^A(b)$  because  $\sim^{eq}$  is transitive.

We conclude that in all three cases,  $(a, b)$  belongs to  $\text{sucs}(\sim^{eq})$ . Therefore,  $\text{sucs}(\sim^{eq})$  is  $\Phi(\sim)$ -closed.

(2) Let  $R_\Sigma$  be the greatest  $\Sigma$ -bisimulation on  $\mathcal{A}$ . Suppose that

$$R_\Sigma^{eq} \text{ is a } \Sigma\text{-bisimulation.} \tag{4}$$

Since  $R_\Sigma$  is the *greatest*  $\Sigma$ -bisimulation, (4) implies  $R_\Sigma^{eq} \subseteq R_\Sigma$ . Since  $R_\Sigma$  is a subset of  $R_\Sigma^{eq}$ , we conclude that both relations agree with each other. Hence  $R_\Sigma$  is an equivalence relation and thus a  $\Sigma$ -congruence.

It remains to show (4), which holds true iff

$$R_{\Sigma}^{eq} \subseteq suc(s(R_{\Sigma}^{eq})). \quad (5)$$

Since  $R_{\Sigma}^{eq} = lfp(\Phi(R_{\Sigma}))$ , fixpoint induction (see chapter 3) implies (5) if  $suc(s(R_{\Sigma}^{eq}))$  is  $\Phi(R_{\Sigma})$ -closed, i.e., if  $\Phi(R_{\Sigma})(suc(s(R_{\Sigma}^{eq}))) \subseteq suc(s(R_{\Sigma}^{eq}))$ .

So let  $(a, b) \in \Phi(R_{\Sigma})(suc(s(R_{\Sigma}^{eq})))$ . Hence  $(a, b) \in suc(s(R_{\Sigma}^{eq}))$  or we have one of the following three cases:

*Case 1:*  $a = b$ . Then for all  $f : e \rightarrow e' \in F$ ,  $f^A(a) = f^A(b)$  and thus  $(f^A(a), f^A(b)) \in R_{\Sigma}^{eq}$  because  $R_{\Sigma}^{eq}$  is reflexive.

*Case 2:*  $(b, a) \in R_{\Sigma}^{eq}$ . Then for all  $f : e \rightarrow e' \in F$ ,  $(f^A(b), f^A(a)) \in R_{\Sigma}^{eq}$ . Hence  $(f^A(a), f^A(b)) \in R_{\Sigma}^{eq}$  because  $R_{\Sigma}^{eq}$  is symmetric.

*Case 3:*  $(a, c), (c, b) \in R_{\Sigma}^{eq}$  for some  $c \in A$ . Then for all  $f : e \rightarrow e' \in F$ ,

$$(f^A(a), f^A(c)), (f^A(c), f^A(b)) \in R_{\Sigma}^{eq}.$$

Hence  $(f^A(a), f^A(c)) \in R_{\Sigma}^{eq}$  because  $R_{\Sigma}^{eq}$  is transitive.

We conclude that in all three cases,  $(a, b)$  belongs to  $suc(s(R_{\Sigma}^{eq}))$ . Therefore,  $suc(s(R_{\Sigma}^{eq}))$  is  $\Phi(R_{\Sigma})$ -closed.

(3)

“ $\Rightarrow$ ”: Let  $h$  be  $\Sigma$ -homomorphic,  $e \in \mathcal{T}_q(S, \mathcal{I})$ ,  $(a, b) \in \text{graph}(h)_e$  and  $f : e \rightarrow e' \in F$ . Then  $h(a) = b$  and thus  $h(f^{\mathcal{A}}(a)) = f^{\mathcal{B}}(h(a)) = f^{\mathcal{B}}(b)$ , i.e.,  $(f^{\mathcal{A}}(a), f^{\mathcal{B}}(b)) \in \text{graph}(h)_{e'}$ .

“ $\Leftarrow$ ”: Let  $\text{graph}(h)$  be a  $\Sigma$ -bisimulation,  $f : e \rightarrow e' \in F$  and  $a \in A_e$ . Then

$$(a, h(a)), (f^{\mathcal{A}}(a), h(f^{\mathcal{A}}(a))) \in \text{graph}(h)$$

and thus  $(f^{\mathcal{A}}(a), f^{\mathcal{B}}(h(a))) \in \text{graph}(h)$ . Hence  $h(f^{\mathcal{A}}(a)) = f^{\mathcal{B}}(h(a))$ . □

## $\Sigma$ -terms and -coterms

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature.

Let  $\Sigma$  be **constructive and polynomial** and  $V$  be an  $S$ -sorted set of “variables”.

$\Sigma$ -terms are trees whose inner nodes are labelled with constructors or (indices of) injections, whose leaves are labelled with indices or variables and whose edges are labelled with (indices of) projections. More precisely:

The set  $\textcolor{red}{CT}_\Sigma(V)$  of (**first-order**)  $\Sigma$ -terms over  $V$  is the **greatest**  $\mathcal{T}_p(S, \mathcal{I})$ -sorted set

$$\textcolor{blue}{M} \subseteq \mathcal{I}^* \rightarrow (\mathcal{I} \cup F \cup V) + 1$$

of labelled trees over  $(\mathcal{I}, F \cup V \cup \mathcal{I})$  such that the following conditions hold true:

- For all  $s \in S$  and  $t \in \textcolor{blue}{M}_s$ ,  $t \in V_s$  or there are  $c : e \rightarrow s \in F$  and  $u \in \textcolor{blue}{M}_e$  such that  $t = c(u)$ . (1)
- For all  $s \subseteq \mathcal{I}$ ,  $\textcolor{blue}{M}_s = s$ . (2)
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t \in \textcolor{blue}{M}_e$  there are  $i \in I$  and  $u \in \textcolor{blue}{M}_{e_i}$  such that  $t = i(u)$ . (3)
- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t \in \textcolor{blue}{M}_e$  there is  $(t_i)_{i \in I} \in \textcolor{blue}{X}_{i \in I} \textcolor{blue}{M}_{e_i}$  such that  $t = ()\{i \rightarrow t_i \mid i \in I\}$ . (4)

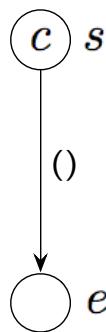
$T_\Sigma(V)$  denotes the least  $\mathcal{T}_p(S, \mathcal{I})$ -sorted set  $M$  of well-founded labelled trees over  $(\mathcal{I}, \mathcal{I} \cup F \cup V)$  such that (2) and the following conditions hold true:

- For all  $c : e \rightarrow s \in F$  and  $u \in M_e$ ,  $c(u) \in M_s$ . (5)

- For all  $s \in S$ ,  $V_s \subseteq M_s$ . (6)

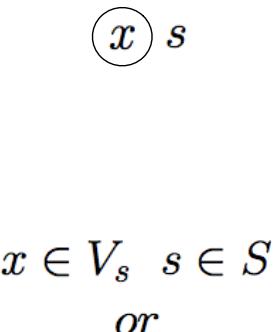
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in M_{e_i}$ ,  $i(t) \in M_e$ . (7)

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $(t_i)_{i \in I} \in \bigtimes_{i \in I} M_{e_i}$ ,  
 $\{\{i \rightarrow t_i \mid i \in I\} \in M_e\}$ . (8)



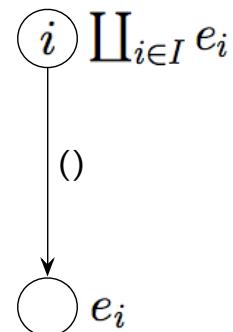
$$c : e \rightarrow s \in F$$

(1/5)

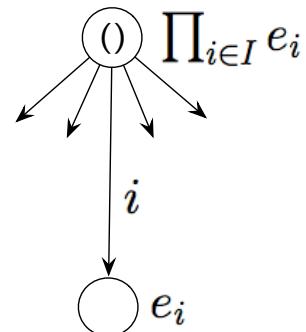


$$x \in V_s \quad s \in S$$

(1/2/6)



(3/7)



(4/8)

For all  $s \in S$ , let  $V_s = \emptyset$ . Then the elements of  $\text{CT}_\Sigma =_{\text{def}} \text{CT}_\Sigma(V)$  und  $\text{T}_\Sigma =_{\text{def}} \text{T}_\Sigma(V)$  are called **ground  $\Sigma$ -terms**.

If for all  $c \in F$ ,  $\text{src}(c)$  does not contain some index set, then for all  $s \in S$ ,  $\text{T}_{\Sigma,s}$  is empty.

Let  $\Sigma$  be **destructive and polynomial** and  $C$  be an  $S$ -sorted set of “colors”.

$\Sigma$ -coterms are trees whose inner nodes are labelled with colors or (indices of) injections, whose leaves are labelled with indices or variables and whose edges are labelled with destructors or (indices of) projections. More precisely:

The set  $\text{DT}_\Sigma(C)$  of  **$\Sigma$ -coterms over  $C$**  is the **greatest**  $\mathcal{T}_p(S, \mathcal{I})$ -sorted set

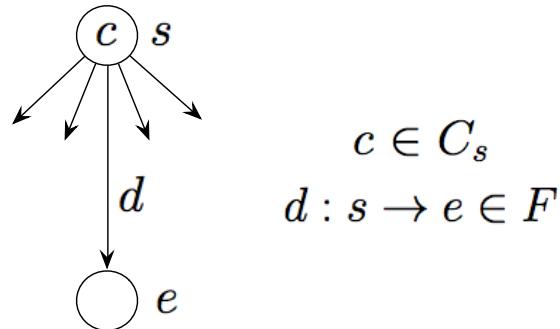
$$M \subseteq (\mathcal{I} \cup F)^* \rightarrow (\mathcal{I} \cup C) + 1$$

of labelled trees over  $(\mathcal{I} \cup F, \mathcal{I} \cup C)$  such that (2), (3) and (4) hold true,

- for all  $s \in S$  and  $t \in M_s$  there are  $c \in C_s$  and  $(t_d)_{d:s \rightarrow e \in F} \in X_{d:s \rightarrow e \in F} M_e$  such that  $t = c\{d \rightarrow t_d \mid d : s \rightarrow e \in F\}$ . (9)

$\text{coT}_\Sigma(C)$  denotes the **least**  $\mathcal{T}_p(S, \mathcal{I})$ -sorted set  $M$  of well-founded labelled trees over  $(\mathcal{I} \cup F, \mathcal{I} \cup C \cup V)$  such that (2), (7) and (8) hold true and

- for all  $s \in S$ ,  $c \in C_s$  and  $(t_d)_{d:s \rightarrow e \in F} \in \textcolor{blue}{X}_{d:s \rightarrow e \in F} M_e$ ,  
 $c\{d \rightarrow t_d \mid d : s \rightarrow e \in F\} \in \textcolor{blue}{M}_s$ . (10)



For all  $s \in S$ , let  $C_s = 1$ . Then the elements of  $DT_\Sigma =_{def} DT_\Sigma(C)$  and  $coT_\Sigma =_{def} coT_\Sigma(C)$  are called **ground  $\Sigma$ -coterms**.

If for all  $d \in D$ ,  $trg(d)$  does not contain some index set, then for all  $s \in S$ ,  $DT_{\Sigma,s}$  is a singleton.

## Sample terms and coterms

Let  $\Sigma = \text{Nat}$ .

$CT_\Sigma$  is the greatest subset  $M$  of  $ltr(1, \{(), zero, succ\})$  with the following property:

- For all  $t \in M$ ,  $t = zero$  or  $t = succ(u)$  for some  $u \in M$ .

$T_\Sigma$  is the least subset  $M$  of  $ltr(1, \{(), zero, succ\})$  with the following properties:

- $zero \in M$ .
- For all  $t \in M$ ,  $succ(t) \in M$ .

Hence  $T_\Sigma \cong \mathbb{N}$  and  $CT_\Sigma \cong \mathbb{N} \cup \{\omega\}$ .

Let  $\Sigma = \text{coNat}$ .

$DT_\Sigma$  is the greatest subset  $M$  of  $ltr(\{()\}, \{(), 1, 2\})$  with the following property:

- For all  $t \in M$ ,  $t = ()\{pred \rightarrow 1\}$  or  $t = ()\{pred \rightarrow 2(u)\}$  for some  $u \in M$ .

Hence  $DT_\Sigma \cong \mathbb{N} \cup \{\omega\}$ .

Let  $\Sigma = \text{List}(X)$ .

$CT_\Sigma$  is the greatest subset  $M$  of  $ltr(\{1, 2\}, \{\alpha, cons\} \cup X)$  with the following property:

- For all  $t \in M$ ,  $t = \alpha$  or  $t = cons(x, u)$  for some  $x \in X$  and  $u \in M$ .

$T_\Sigma$  is the least subset  $M$  of  $ltr(\{1, 2\}, \{\alpha, cons\} \cup X)$  with the following properties:

- $\alpha \in M$ .
- For all  $x \in X$  and  $t \in M$ ,  $cons(x, t) \in M$ .

Hence  $T_\Sigma \cong X^*$  and  $CT_\Sigma \cong X^\infty = X^* \cup X^{\mathbb{N}}$ .

Let  $\Sigma = \text{coList}(X)$ .

$DT_\Sigma$  is the greatest subset  $M$  of  $ltr(\{1, 2\}, 1 \cup X)$  with the following property:

- For all  $t \in M$ ,  $t = ()\{split \rightarrow 1\}$  or  $t = ()\{split \rightarrow 2(x, u)\}$  for some  $x \in X$  and  $u \in M$ .

Hence  $DT_\Sigma \cong X^\infty = X^* \cup X^{\mathbb{N}}$ .

Let  $\Sigma = \text{Reg}(X)$ .

$T_\Sigma$  is the least subset  $M$  of  $ltr(\{1, 2\}, \{\text{par}, \text{seq}, \text{iter}, \text{eps}, \text{base}\} \cup \mathcal{P}(X))$  with the following properties:

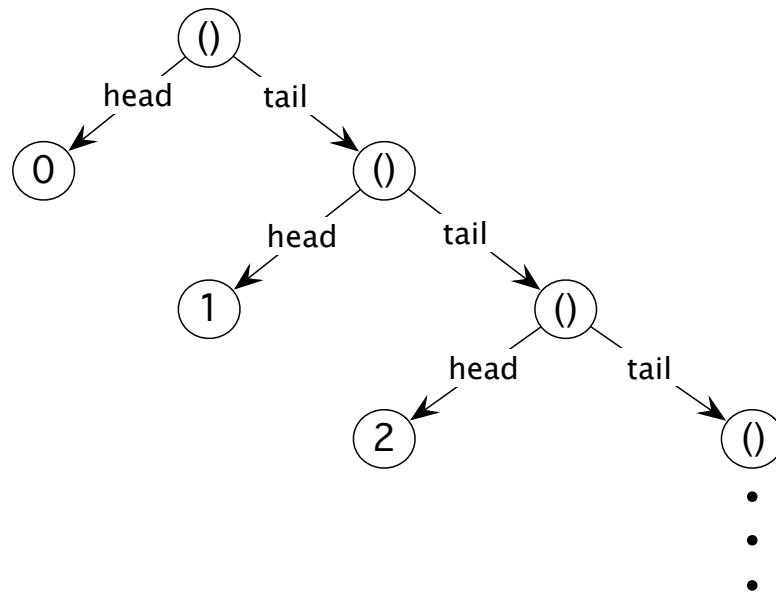
- For all  $t, u \in M$  and  $B \subseteq X$ ,  $\text{par}(t, u)$ ,  $\text{seq}(t, u)$ ,  $\text{iter}(t)$ ,  $\text{eps}$ ,  $\text{base}(B) \in M$ .

Let  $\Sigma = \text{Stream}(X)$ .

$DT_\Sigma$  is the greatest subset  $M$  of  $ltr(\{\text{head}, \text{tail}\}, 1 \cup X)$  with the following property:

- For all  $t \in M$ ,  $t = ()\{\text{head} \rightarrow x, \text{tail} \rightarrow u\} \in M$  for some  $x \in X$  and  $u \in M$ .

Hence  $DT_\Sigma \cong X^{\mathbb{N}}$ .



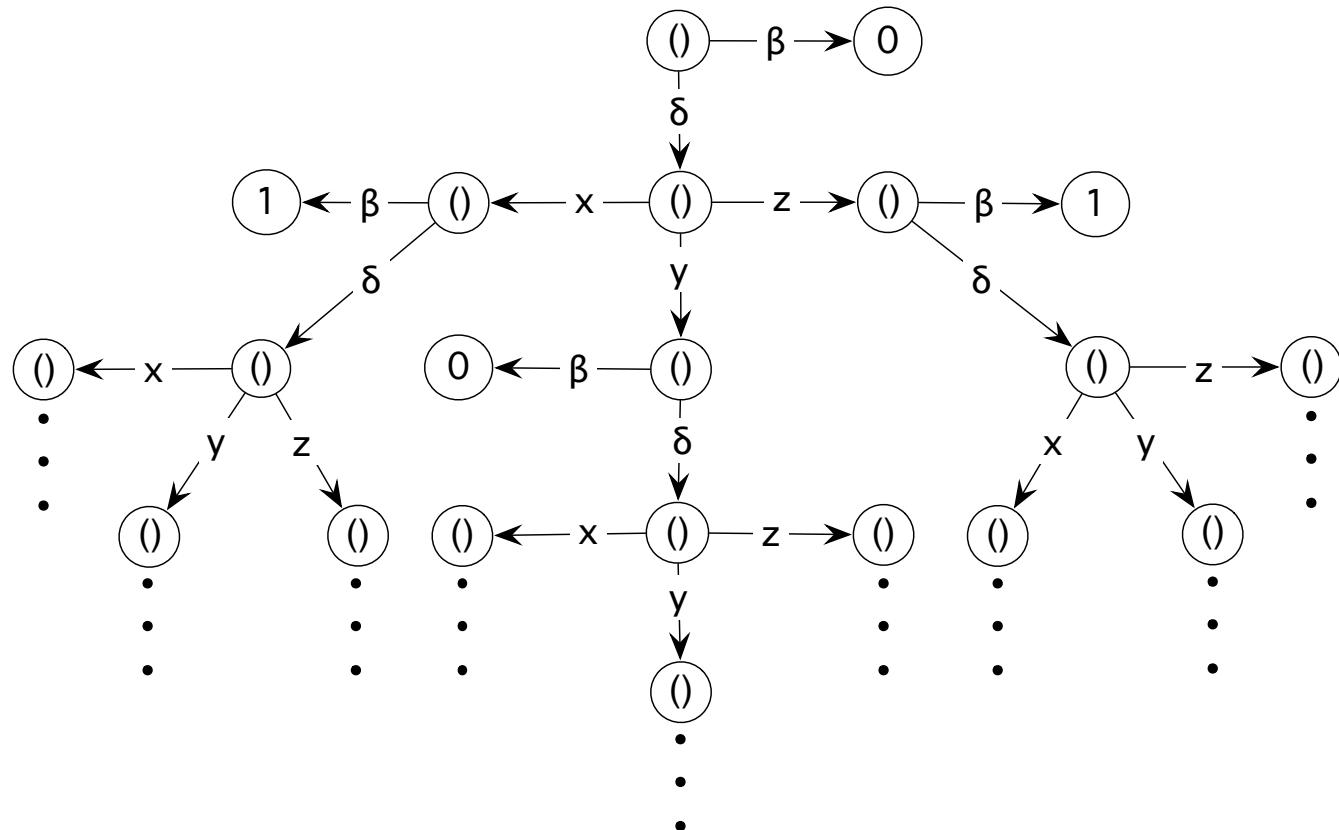
*Stream( $\mathbb{N}$ )-coterm that represents the stream of natural numbers.*

Let  $\Sigma = DAut(X, Y)$ .

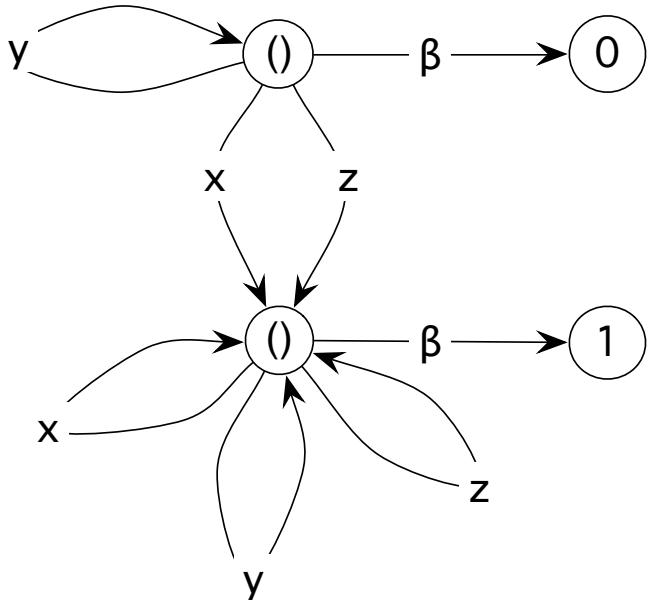
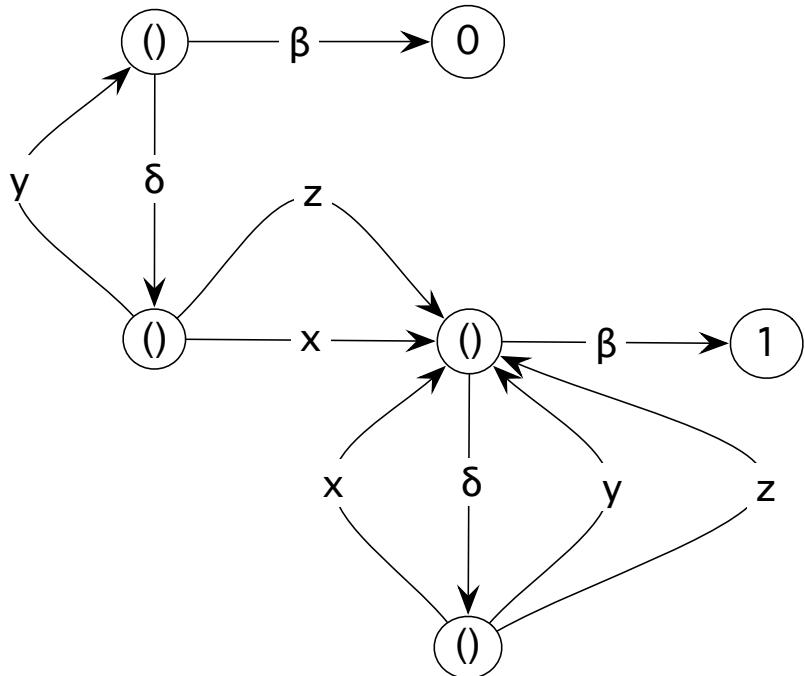
$DT_\Sigma$  is the greatest subset  $M$  of  $ltr(\{\delta, \beta\} \cup X, 1 \cup Y)$  with the following property:

- For all  $t \in M$ ,  $t = ()\{\delta \rightarrow ()\{x \rightarrow t_x \mid x \in X\}, \beta \rightarrow y\}$  for some  $(t_x)_{x \in X} \in M^X$  and  $y \in Y$ .

Hence  $DT_\Sigma \cong Y^{X^*}$ .



$DAut(\{x, y, z\}, 2)$ -coterm representing an acceptor of all words over  $\{x, y, z\}$   
that contain  $x$  or  $z$



Folding of the above infinite, but rational cotermin into a finite graph (left)  
and the corresponding transition graph (right)

## Term and cotype algebras

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature and  $V \in Set^S$ .

$CT_\Sigma(V)$  is a  $\Sigma$ -algebra:

- For all  $c : e \rightarrow s \in C$  and  $t \in CT_\Sigma(V)_e$ ,  $c^{CT_\Sigma(V)}(t) =_{def} c(t)$ .
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in CT_\Sigma(V)_{e_i}$ ,

$$CT_\Sigma(V)_e =_{def} \{i(t) \mid i \in I, t \in CT_\Sigma(V)_{e_i}\},$$

$$\iota_i(t) =_{def} i(t).$$

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $(\lambda i \rightarrow t_i \mid i \in I) \in CT_\Sigma(V)_e$ ,

$$CT_\Sigma(V)_e =_{def} \{(\lambda i \rightarrow t_i \mid i \in I) \mid (t_i)_{i \in I} \in \bigtimes_{i \in I} CT_\Sigma(V)_{e_i}\},$$

$$\pi_i((\lambda i \rightarrow t_i \mid i \in I)) =_{def} t_i.$$

Hence the carrier of  $CT_\Sigma(V)$  is a model of  $\mathcal{T}_p(S, \mathcal{I})$ .

Moreover,  $T_\Sigma(V)$  is a  $\Sigma$ -subalgebra of  $CT_\Sigma(V)$ .

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature and  $V \in Set^S$ .

$DT_\Sigma(C)$  is a  $\Sigma$ -algebra:

- For all  $d : s \rightarrow e \in D$  and  $t \in DT_\Sigma(C)_s$ ,  $d^{DT_\Sigma(C)}(t) =_{def} \lambda w.t(dw)$ .
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in DT_\Sigma(C)_{e_i}$ ,

$$DT_\Sigma(C)_e =_{def} \{i(t) \mid i \in I, t \in DT_\Sigma(C)_{e_i}\},$$

$$\iota_i(t) =_{def} i(t).$$

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $()\{i \rightarrow t_i \mid i \in I\} \in DT_\Sigma(C)_e$ ,

$$DT_\Sigma(C)_e =_{def} \{()\{i \rightarrow t_i \mid i \in I\} \mid (t_i)_{i \in I} \in \times_{i \in I} DT_\Sigma(C)_{e_i}\},$$

$$\pi_i(()\{i \rightarrow t_i \mid i \in I\}) =_{def} t_i.$$

Hence the carrier of  $DT_\Sigma(C)$  is a model of  $\mathcal{T}_p(S, \mathcal{I})$ .

Moreover,  $coT_\Sigma(C)$  is a  $\Sigma$ -subalgebra of  $DT_\Sigma(C)$ .

By the interpretation of destructors in  $DT_\Sigma(C)$ , coterms become a kind of analytic functions: Two coterms  $t, t' \in DT_\Sigma(C)_s$  are equal if and only if the “initial values”  $t(\epsilon)$  and  $t'(\epsilon)$  and for all  $d : s \rightarrow e$  the “derivatives”  $d^{DT_\Sigma(C)}(t)$  and  $d^{DT_\Sigma(C)}(t')$  coincide.

## Term folding (“operational semantics”)

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature,  $V \in Set^S$  and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .

A **term valuation** of  $V$  in  $A$  is an  $S$ -sorted function  $\textcolor{blue}{g} : V \rightarrow A$ . From now on,  $\textcolor{red}{A}^V$  denotes the set of term valuations of  $V$  in  $A$ .

Given  $g \in A^V$ , the **term extension**  $\textcolor{blue}{g}^* : T_\Sigma(V) \rightarrow A$  of  $g$  to  $T_\Sigma(V)$ , also called **term folding**, is the  $\mathcal{T}_p(S, \mathcal{I})$ -sorted function that is defined inductively as follows:

- For all  $s \in S$  and  $x \in V_s$ ,  $\textcolor{blue}{g}_s^*(x) = g_s(x)$ . (1)

- For all  $s \subseteq \mathcal{I}$  and  $a \in s$ ,  $\textcolor{blue}{g}_s^*(a) = \textcolor{blue}{a}$ . (2)

- For all  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(V)_e$ ,  $\textcolor{blue}{g}_s^*(c(t)) = c^A(g_e^*(t))$ . (3)

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in T_\Sigma(V)_{e_i}$ ,

$$\textcolor{blue}{g}_e^*(i(t)) = \iota_i(g_{e_i}^*(t)). \quad (4)$$

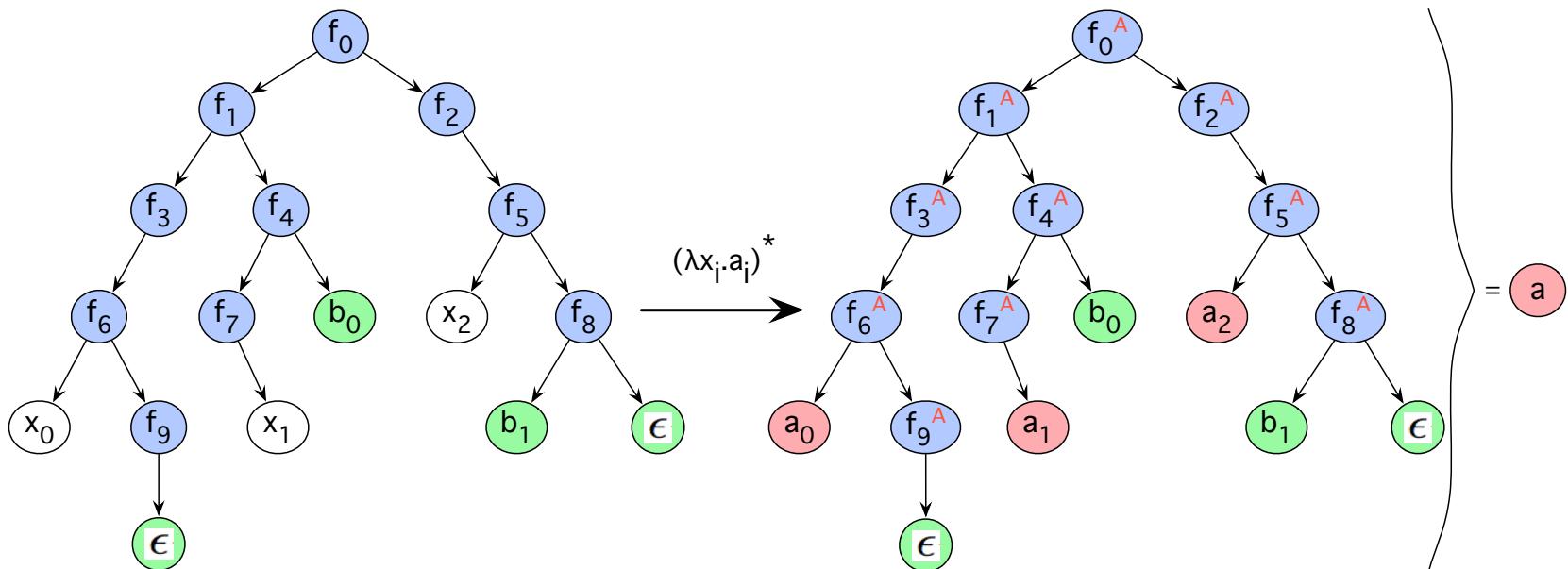
- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $t = ()\{i \rightarrow t_i \mid i \in I\} \in T_\Sigma(V)_e$  and  $i \in I$ ,

$$\pi_i(g_e^*(t)) = g_{e_i}^*(t_i). \quad (5)$$

Note that (2), (4) and (5) follow from the lifting of  $g^*$  as an  $S$ -sorted function to a  $\mathcal{T}_p(S, \mathcal{I})$ -sorted function (see chapter 7).

Intuitively,  $g^*$  evaluates each well-founded  $\Sigma$ -term over  $V$  in  $\mathcal{A}$ .

In particular,  $\text{id}_A^* : T_\Sigma(A) \rightarrow \mathcal{A}$  interprets the constructors and injections of  $t \in T_\Sigma(A)$  bottom-up, starting at the leaves of  $t$  that are labelled with elements of  $A$ . For all  $c : e \rightarrow s \in C$  and  $a \in A_e$ ,  $\text{id}_A^*(c(a)) = c^{\mathcal{A}}(a)$ .



*Illustration of folding*

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $e \in \mathcal{T}_p(S, \mathcal{I})$ .  $t \in T_\Sigma(V)_e$  and  $\mathcal{A}$  define the function  $t^{\mathcal{A}} : A^V \rightarrow A_e$ : For all  $g \in A^V$ ,

$$t^{\mathcal{A}}(g) =_{def} g^*(t).$$

Given  $s \in S$  and  $t, t' \in T_\Sigma(V)_s$ ,  $g \in A^V$  solves the (**first-order**) equation  $t = t'$  in  $\mathcal{A}$ , written as  $\mathcal{A} \models_g t = t'$ , if  $g^*(t) = g^*(t')$ , or, equivalently,  $t^{\mathcal{A}} = t'^{\mathcal{A}}$ .

$\mathcal{A}$  satisfies the (**first-order**) **conditional equation**  $\bigwedge_{i=1}^n t_i = t'_i \Rightarrow t = t'$  if every  $g \in A^V$  that solves  $t_1 = t'_1, \dots, t_n = t'_n$  also solves  $t = t'$ .

An empty conjunction is identified with *True* and mostly omitted, i.e., a conditional equation  $\text{True} \Rightarrow t = t'$  is simply written as  $t = t'$ .

Consequently,  $\mathcal{A}$  satisfies  $t = t'$  if all  $g \in A^V$  solve  $t = t'$ .

If  $\mathcal{A} = T_\Sigma(V')$  for some  $V' \in \text{Set}^S$ , then  $g$  is called a **term substitution** because  $g^* : T_\Sigma(V) \rightarrow T_\Sigma(V')$  replaces the variables of a term by terms: For all  $x \in V$ ,  $x$  is replaced by  $g(x)$ .

## Theorem FREE

Let  $V \in Set^S$ . Then  $T_\Sigma(V)$  is a **free  $\Sigma$ -algebra over  $V$** , i.e., for all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$  and  $g \in A^V$ ,  $g^*$  is the only  $\Sigma$ -homomorphism from  $T_\Sigma(V)$  to  $\mathcal{A}$  that satisfies (5).

$$\begin{array}{ccc}
 V & \xrightarrow{\text{inc}_V} & T_\Sigma(V) \\
 & \searrow g \quad (5) \quad \nearrow g^* & \\
 & A &
 \end{array}$$

In particular, if for all  $s \in S$ ,  $V_s = \emptyset$ , then there is exactly one term valuation  $g \in A^V$ , (5) reduces to the uniqueness of  $g^*$  and thus  $T_\Sigma = T_\Sigma(V)$  is initial in  $Alg_\Sigma$ .  $g^*$  no longer depends on  $g$  and is denoted by  $\text{fold}^\mathcal{A}$ . This notation is also used for the restriction of  $g^*$  to  $T_\Sigma$  if  $V$  is nonempty.

*Proof.* By (1),  $g^*$  satisfies (5).

$g^*$  is  $\Sigma$ -homomorphic: For all  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(V)_e$ ,

$$g_s^*(c^{T_\Sigma(V)}(t)) = g_s^*(c(t)) \stackrel{(2)}{=} c^\mathcal{A}(g_e^*(t)).$$

$g^*$  is unique: Let  $h : T_\Sigma(V) \rightarrow \mathcal{A}$  be a  $\Sigma$ -homomorphism with  $h \circ inc_V = g$ .

- For all  $s \in S$  and  $x \in V_s$ ,  $g_s^*(x) = g_s(x) \stackrel{h \circ inc_V = g}{=} h_s(x)$ .

- For all  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(V)_e$ ,

$$g_s^*(c(t)) \stackrel{(2)}{=} c^{\mathcal{A}}(g_e^*(t)) \stackrel{ind. \ hyp.}{=} c^{\mathcal{A}}(h_e(t)) \stackrel{h \ hom.}{=} h_s(c^{T_\Sigma(V)}(t)) = h_s(c(t)).$$

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in T_\Sigma(V)_{e_i}$ ,

$$g_e^*(i(t)) \stackrel{(3)}{=} \iota_i(g_{e_i}^*(t)) \stackrel{ind. \ hyp.}{=} \iota_i(h_{e_i}(t)) = h_e(\iota_i(t)) = h_e(i(t)).$$

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $(t_i)_{i \in I} \in \bigtimes_{i \in I} T_\Sigma(V)_{e_i}$ ,

$$\begin{aligned} \pi_i(g_e^*((\lambda i \rightarrow t_i \mid i \in I))) &\stackrel{(4)}{=} g_{e_i}^*(t_i) \stackrel{ind. \ hyp.}{=} h_{e_i}(t_i) = h_{e_i}(\pi_i((\lambda i \rightarrow t_i \mid i \in I))) \\ &= \pi_i(h_e((\lambda i \rightarrow t_i \mid i \in I))). \end{aligned}$$

Hence for all  $e \in \mathcal{T}_p(S, \mathcal{I})$ ,  $g_e^* = h_e$ . □

Exercise 11 Show  $fold^{Bool} = \beta^{Bro(X)}$ .

Exercise 12 Show by structural induction that for all  $t \in T_{Reg(X)}$ ,

$$\epsilon \in fold^{Lang(X)}(t) \iff fold^{Bool}(t) = 1.$$

## Lemma SUBST

For all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$ ,  $g \in A^V$  and  $\Sigma$ -homomorphisms  $h : \mathcal{A} \rightarrow \mathcal{B}$ ,

$$(h \circ g)^* = h \circ g^*.$$

*Proof.* Since  $h \circ g^* \circ inc_V = h \circ g$ , the conjecture follows from the fact that  $(h \circ g)^*$  is the only  $\Sigma$ -homomorphism  $h' : T_\Sigma(V) \rightarrow \mathcal{B}$  with  $h' \circ inc_V = h \circ g$ .  $\square$

Since  $g^*$  is the only  $\Sigma$ -homomorphism from  $T_\Sigma(V)$  to  $\mathcal{A}$  that satisfies (5),  $fold^\mathcal{A}$  is the only  $\Sigma$ -homomorphism from  $T_\Sigma$  to  $\mathcal{A}$ , i.e.,  **$T_\Sigma$  is initial in  $Alg_\Sigma$** .

A  $\Sigma$ -algebra  $\mathcal{A}$  is **equationally consistent** if  $fold^\mathcal{A}$  is mono.

$\mathcal{A}$  is **reachable** (or **generated**) if  $fold^\mathcal{A}$  is epi.

**$RAlg_\Sigma$**  denotes the full subcategory of  $Alg_\Sigma$  whose objects are all reachable  $\Sigma$ -algebras.

By Lemma KER (2), for all  $\mathcal{A} \in RAlg_\Sigma$ ,  $\mathcal{A} \cong T_\Sigma / ker(fold^\mathcal{A})$ .

## Lemma INIREACH

Let  $\mathcal{K}$  be a full subcategory of  $RAlg_\Sigma$  and the  $\Sigma$ -algebra  $\mathcal{B} = \mathcal{B}(\mathcal{K})$  be defined as follows:

- For all  $s \in S$ ,  $R_s = \ker(\langle fold^{\mathcal{A}} \rangle_{\mathcal{A} \in \mathcal{K}})_s = \bigcap_{\mathcal{A} \in \mathcal{K}} \ker(fold^{\mathcal{A}})_s$  (see [product equation 9](#)) and  $\mathcal{B}(s) = T_{\Sigma,s}/R_s$ .
- For all  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma$ ,  $c^{\mathcal{B}}(nat_{R,e}(t)) = nat_{R,s}(c(t))$ .

For all  $\mathcal{A} \in \mathcal{K}$ , there is a unique  $\Sigma$ -homomorphism from  $\mathcal{B}$  to  $\mathcal{A}$ .

In particular,  $\mathcal{B}$  is initial in  $RAlg_\Sigma$ .

*Proof.* By [Lemma KER](#) (2), there is a unique  $\Sigma$ -monomorphism  $h : \mathcal{B} \rightarrow \prod \mathcal{K}$  such that  $h \circ nat_R = \langle fold^{\mathcal{A}} \rangle_{\mathcal{A} \in \mathcal{K}}$ . Hence for all  $\mathcal{A} \in \mathcal{K}$ ,  $\pi_{\mathcal{A}} \circ h : \mathcal{B} \rightarrow \mathcal{A}$  is  $\Sigma$ -homomorphic. Suppose that there are two  $\Sigma$ -homomorphisms  $h_1, h_2 : \mathcal{B} \rightarrow \mathcal{A}$ . Since  $T_\Sigma$  is initial in  $Alg_\Sigma$ ,  $h_1 \circ nat_R = h_2 \circ nat_R$ . Hence  $h_1 = h_2$  because  $nat_R$  is epi.

In particular, since  $\mathcal{B} \in RAlg_\Sigma$ ,  $\mathcal{B}$  is initial in  $RAlg_\Sigma$ . □

## Example

Let  $\Sigma = Dyn(X, 1)$ ,  $\mathcal{C}$  be a  $coStream(X)$ -algebra and  $\mathcal{K}$  be the category of reachable  $\Sigma$ -algebras  $\mathcal{A}$  with  $\mathcal{A}|_{coStream(X)} = \mathcal{C}$ . Then  $\mathcal{B}(\mathcal{K})$  agrees with the free (1-)pointed automaton over  $\mathcal{C}$  as defined in [149], section 5, and a quotient of the initial reachable  $\Sigma$ -algebra. □

## Term grounding

Let  $\mathcal{I}(V) = \mathcal{I} \cup V$ ,  $\mathcal{C}(V) = C \cup \{val_s : V_s \rightarrow s \mid s \in S\}$  and

$$\Sigma(V) = (S, \mathcal{I}(V), \mathcal{C}(V))$$

is called the **grounding of  $\Sigma$  on  $V$** .

$T_\Sigma(V)$  is a  $\Sigma(V)$ -algebra: For all  $s \in S$  and  $x \in V_s$ ,  $val_s^{T_\Sigma(V)}(x) =_{def} x$ .

Let  $\mathcal{A}$  be a  $\Sigma(V)$ -algebra with carrier  $A$ . Since

$$(val^\mathcal{A})^* \circ val^{T_\Sigma(V)} = (val^\mathcal{A})^* \circ inc_V = val^\mathcal{A},$$

$(val^\mathcal{A})^*$  is compatible with  $val$  and thus  $\Sigma(V)$ -homomorphic.

Vice versa, two  $\Sigma(V)$ -homomorphisms  $h, h' : T_\Sigma(V) \rightarrow \mathcal{A}$  are compatible with  $val$ . Hence

$$h \circ inc_V = h \circ val^{T_\Sigma(V)} = val^\mathcal{A} = h' \circ val^{T_\Sigma(V)} = h' \circ inc_V.$$

Since  $h$  and  $h'$  are  $\Sigma$ -homomorphic, we conclude  $h = h'$ .

Hence  $T_\Sigma(V)$  is initial in  $Alg_{\Sigma(V)}$  and for all  $\mathcal{A} \in Alg_{\Sigma(V)}$ ,

$$fold^\mathcal{A} = (val^\mathcal{A})^*.$$

By replacing the label  $x \in V$  of every leaf  $n$  of  $t \in T_\Sigma(V)$  with  $val$  and adding an edge to  $t$  with source  $n$ , label  $\epsilon$  and a new target node labelled with  $x$ , we obtain a  $\Sigma(V)$ -isomorphism from  $T_\Sigma(V)$  to  $T_{\Sigma(V)}$ .

Moreover,  $H_{\Sigma(V)} = H_\Sigma + V$  (see  $\Sigma$ -functors).

## Sample initial algebras

Since initial algebras are unique up to isomorphism,

- $T_{Nat} \cong \mathbb{N}$  is initial in  $Alg_{Nat}$  (see sample algebra 1), (1)

- $T_{Dyn(X,Y)} \cong Seq(X, Y)$  is initial in  $Alg_{Dyn(X,Y)}$  (see sample algebra 3), (2)

- $T_{List(X)} \cong X^*$  is initial in  $Alg_{List(X)}$  (see sample algebra 3), (3)

- $T_{coStream(X)} \cong \emptyset$  is initial in  $Alg_{coStream(X)}$ ,

- $T_{Bintree(X)} \cong FBin(X)$  is initial in  $Alg_{Bintree(X)}$  (see sample algebra 10), (4)

- $T_{Tree(X)} \cong FTree(X)$  is initial in  $Alg_{Tree(X)}$  (see sample algebra 13), (5)

- $T_{Reg(X)}$  is initial in  $Alg_{Reg(X)}$ . (6)

Given a constructive signature  $\Sigma$  and a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$ ,  $fold^{\mathcal{A}}$  denotes the unique  $\Sigma$ -homomorphism not only from  $T_\Sigma$  to  $\mathcal{A}$ , but also from isomorphic representations of  $T_\Sigma$  like those listed above.

In cases (1)-(5), the respective inductive (!) definition of  $h =_{def} fold^{\mathcal{A}}$  reads as follows:

## 1. *Nat*-algebra $\mathbb{N}$

$$\begin{aligned} h : \mathbb{N} &\rightarrow A \\ 0 &\mapsto \text{zero}^A \\ n + 1 &\mapsto \text{succ}^A(h(n)) \end{aligned}$$

## 2. *Dyn*( $X, Y$ )-algebra $\text{Seq}(X, Y)$

$$\begin{aligned} h : X^* \times Y &\rightarrow A \\ (\epsilon, y) &\mapsto \alpha^A(y) \\ (xw, y) &\mapsto \text{cons}^A(x, h(w, y)) \end{aligned}$$

For all  $B \subseteq A$ ,  $(\mathcal{A}, B)$  **realizes**  $f \in Y^{X^*}$  if for all  $y \in Y$ ,

$$h(w, y) \in B \Leftrightarrow f(w) = y.$$

Let  $Y = 1$ . For all  $B \subseteq A$ ,  $(\mathcal{A}, B)$  **accepts** the **language**  $L \subseteq X^*$  if

$$h(w, ()) \in B \Leftrightarrow w \in L.$$

### 3. $List(X)$ -algebra $X^*$

$$\begin{aligned} h : X^* &\rightarrow A \\ \epsilon &\mapsto \alpha^A \\ x \cdot w &\mapsto cons^A(x, h(w)) \end{aligned}$$

$h$  is  $List(X)$ -homomorphic:

$$\begin{aligned} h(\alpha^{X^*}) &= h(\epsilon) = \alpha^A, \\ h(cons^{X^*}(x, w)) &= h(x \cdot w) = cons^A(x, h(w)). \end{aligned}$$

$h$  is the only  $List(X)$ -homomorphism from  $X^*$  to  $\mathcal{A}$ :

Let  $h' : X^* \rightarrow A$  be  $List(X)$ -homomorphic. Then

$$\begin{aligned} h'(\epsilon) &= h(\alpha^{X^*}) \stackrel{h' \text{ hom.}}{=} \alpha^A = h(\epsilon), \\ h'(x \cdot w) &= h'(cons^{X^*}(x, w)) \stackrel{h' \text{ hom.}}{=} cons^A(x, h'(w)) \stackrel{\text{ind. hyp.}}{=} cons^A(x, h(w)) = h(x \cdot w), \end{aligned}$$

i.e.,  $h' = h$ .

**Exercise 13** Show that

$$\begin{aligned} length : X^* &\rightarrow \mathbb{N} \\ \epsilon &\mapsto 0 \\ x \cdot w &\mapsto 1 + length(w) \end{aligned}$$

is  $List(X)$ -homomorphic and thus agrees with  $fold^{Length}$  (see sample algebra 1).

#### 4. $Bintree(X)$ -algebra $FBin(X)$

$$\begin{aligned}
 h : ftr(2, X) &\rightarrow A \\
 \Omega &\mapsto \text{empty}^A \\
 x\{0 \rightarrow t, 1 \rightarrow u\} &\mapsto bjoin^A(x, h(t), h(u))
 \end{aligned}$$

#### 5. $Tree(X)$ -algebra $FTree(X)$

$$\begin{aligned}
 h_{tree} : otr(\mathbb{N}, <, X) \cap ftr(\mathbb{N}, X) &\rightarrow A_{tree} \\
 x &\mapsto join^A(x, \text{nil}^A) \\
 x(t_1, \dots, t_n) &\mapsto join^A(x, h_{trees}(t_1, \dots, t_n)) \\
 h_{trees} : (otr(\mathbb{N}, <, X) \cap ftr(\mathbb{N}, X))^* &\rightarrow A_{trees} \\
 \epsilon &\mapsto \text{nil}^A \\
 t \cdot ts &\mapsto cons^A(h_{tree}(t), h_{trees}(ts))
 \end{aligned}$$

#### 6. $Reg(X)$ -algebra $T_{Reg(X)}$

For all  $t \in T_{Reg(X)}$ ,  $L(t) =_{def} fold^{Lang(X)}(t)$  (see example 17 above) is called the **language** of  $t$  that is accepted by, for instance,  $(Bro(X), t)$  (see Sample final algebras).

**Exercise 14** Let  $Bool$  be the  $\Sigma$ -algebra of example 20 above. Given  $t \in T_{\Sigma, state}$ , show that  $L(t)$  contains  $\epsilon$  iff  $fold^{Bool}(t) = 1$ . □

Context-free grammars with base languages (For more details, see [125].)

A **context-free grammar** (CFG)  $G = (S, X, R)$  consists of

- a finite set  $S$  of sorts,
- a set  $X$  of constants—whose subsets are called **base languages**,
- a finite set  $R$  of **rules**  $s \rightarrow w$  with  $s \in S$  and  $w \in (S \cup \mathcal{P}(X))^*$ .

$S$  is the set of **nonterminals**.

The singletons of  $\mathcal{P}(X)$  form the set  $Z(G)$  of **terminals** and usually written without curly brackets.

The following function  $\text{type} : (S \cup \mathcal{P}(X))^* \rightarrow \mathcal{T}_p(S, \mathcal{P}(X))$  removes all elements of  $Z(G)$  from words and translates the latter into the corresponding product types:

- $\text{type}(\epsilon) = 1$ .
- For all  $x \in Z(G)$  and  $w \in (S \cup \mathcal{P}(X))^*$ ,  $\text{type}(x \cdot w) = \text{type}(w)$ .
- For all  $s \in S \cup \mathcal{P}(X) \setminus Z(G)$  and  $w \in (S \cup \mathcal{P}(X))^*$ ,  $\text{type}(sw) = s \times \text{type}(w)$ .

The constructive signature

$$\Sigma(G) = (S, X, \{f_{s \rightarrow w} : type(w) \rightarrow s \mid s \rightarrow w \in R\})$$

is called the **abstract syntax of  $G$** .

Finite ground  $\Sigma(G)$ -terms are called **syntax trees of  $G$** .

The **word algebra of  $G$** ,  $Word(G)$ , recovers the concrete syntax from the abstract one:

- For all  $s \in S$ ,  $Word(G)_s =_{def} X^*$ .
- For all  $w_0 \dots w_n \in Z(G)^*$ ,  $s_1, \dots, s_n \in (S \cup \mathcal{P}(X)) \setminus Z(G)$ ,  $r = (s \rightarrow w_0 e_1 w_1 \dots e_n w_n) \in R$  and  $(v_1, \dots, v_n) \in Word(G)_{s_1 \times \dots \times s_n} = (X^*)^n$ ,

$$f_r^{Word(G)}(v_1, \dots, v_n) =_{def} w_0 v_1 w_1 \dots v_n w_n.$$

The **language  $L(G)$  of  $G$**  is the set of words over  $X$  that result from folding syntax trees in  $Word(G)$ , i.e.,

$$L(G) =_{def} img(fold^{Word(G)}).$$

## Example

The grammar  $\text{SAB}$  consists of the sorts  $S$ ,  $A$  and  $B$ , the terminals  $a$  and  $b$  and the rules

$$\begin{array}{lll} r_1 = S \rightarrow aB, & r_2 = S \rightarrow bA, & r_3 = S \rightarrow \epsilon, \\ r_4 = A \rightarrow aS, & r_5 = A \rightarrow bAA, & r_6 = B \rightarrow bS, \quad r_7 = B \rightarrow aBB. \end{array}$$

Hence  $\Sigma(\text{SAB})$  has the following constructors:

$$\begin{array}{lll} f_1 : B \rightarrow S, & f_2 : A \rightarrow S, & f_3 : 1 \rightarrow S, \\ f_4 : S \rightarrow A, & f_5 : A \times A \rightarrow A, & f_6 : S \rightarrow B, \quad f_7 : B \times B \rightarrow B. \end{array}$$

The word algebra  $\mathcal{W}$  of  $\text{SAB}$  is defined as follows:

$$\mathcal{W}_S = \mathcal{W}_A = \mathcal{W}_B = \{a, b\}^*,$$

and for all  $v, w \in \{a, b\}^*$ ,

$$\begin{aligned} f_1^{\mathcal{W}}(w) &= f_4^{\mathcal{W}}(w) = aw, \\ f_2^{\mathcal{W}}(w) &= f_6^{\mathcal{W}}(w) = bw, \\ f_3^{\mathcal{W}} &= \epsilon, \\ f_5^{\mathcal{W}}(v, w) &= bvw, \\ f_7^{\mathcal{W}}(v, w) &= avw. \end{aligned}$$

$$L(G)_S = \{w \in \{a, b\}^* \mid \#a(w) = \#b(w)\}. \quad (1)$$

A  $\Sigma$ (SAB)-algebra  $\mathcal{A}$  is defined as follows:

$$\begin{aligned} \mathcal{A}_S &= \mathcal{A}_A = \mathcal{A}_B = \mathbb{N}^2, \\ f_1^{\mathcal{A}} &= f_4^{\mathcal{A}} = \lambda(i, j).(i + 1, j), \\ f_2^{\mathcal{A}} &= f_6^{\mathcal{A}} = \lambda(i, j).(i, j + 1), \\ f_3^{\mathcal{A}} &= (0, 0), \\ f_5^{\mathcal{A}} &= \lambda((i, j), (k, l)).(i + k, j + l + 1), \\ f_7^{\mathcal{A}} &= \lambda((i, j), (k, l)).(i + k + 1, j + l). \end{aligned}$$

### Exercise 15

Show  $fold^{\mathcal{A}} = \langle \#a, \#b \rangle \circ fold^{\mathcal{W}}$ ,  $img(fold_S^{\mathcal{A}}) = \Delta_{\mathbb{N}}$ ,  $img(fold_A^{\mathcal{A}}) = \{(i + 1, i) \mid i \in \mathbb{N}\}$  and  $img(fold_B^{\mathcal{A}}) = \{(i, i + 1) \mid i \in \mathbb{N}\}$  and derive (1) from these equations.

For other proofs of (1), see [Iterative equations of a CFG](#) or [125], Beispiele 16.4. □

According to [125], generic compilers for  $G$  can be formulated in category-theoretic terms as follows:

Let  $(M : Set_b^S \rightarrow Set_b^S, \eta, \epsilon)$  be a monad that encapsulates the compiler output or, in the case of incorrect input, returns error messages,  $\mathcal{P} : Set_b^S \rightarrow Set_b^S$  be the ( $S$ -sorted) powerset functor,  $M \times M = \_ \times \_ \circ \Delta \circ M$ ,

$$\oplus : M \times M \rightarrow M \quad \text{and} \quad set : M \rightarrow \mathcal{P}$$

be natural transformations and

$$E_M = \{m \in M(A) \mid A \in Set_b^S, set(m) = \emptyset\}$$

such that for all sets  $A$  and  $B$ ,  $m, m', m'' \in M(A)$ ,  $e \in E$ ,  $f : A \rightarrow M(B)$ ,  $h : A \rightarrow B$  and  $a \in A$ ,

$$\begin{aligned} (m \oplus m') \oplus m'' &= m \oplus (m' \oplus m''), \\ M(h)(e) &= e, \\ M(h)(m \oplus m') &= M(h)(m) \oplus M(h)(m'), \\ set_A(m \oplus m') &\subseteq set_A(m) \cup set_A(m'), \\ set_A(\eta_A(a)) &= \{a\}, \\ set_B(m \gg f) &= \bigcup \{set_B(f(a)) \mid a \in set_A(m)\}. \end{aligned}$$

As a functor from  $Alg_{\Sigma(G)}$  to  $Set$ ,  $X^*$  maps an algebra to  $X^*$  and a homomorphism to  $id_{X^*}$ . Let  $\mathcal{U}$  be the forgetful functor from  $Alg_{\Sigma(G)}$  to  $Set$  that maps an algebra to the union of its carriers and a homomorphism to its underlying function. Let  $\mathcal{W} = Word(G)$ .

A natural transformation  $compile_G : const(X^*) \rightarrow MU$  is a **generic compiler for  $G$**  if  $set_W \circ compile_G^W$  is the following sum extension:

$$\begin{array}{ccccc}
 L(G) & \xrightarrow{inc_{L(G)}} & X^* & \xleftarrow{inc_{X^* \setminus L(G)}} & X^* \setminus L(G) \\
 & \searrow \lambda w.\{w\} & \downarrow set_W \circ compile_G^W & \swarrow \lambda w.\emptyset & \\
 & & \mathcal{P}(X^*) & &
 \end{array}$$

$compile_G$  is parameterized both with a  $\Sigma(G)$ -algebra that represents a target language and a monad, which determines which target objects or error messages are returned.

Following the classical notion of compiler correctness [104, 165], we call  $compile_G^A$  **correct w.r.t. a source model  $Sem$  and a target model  $Mach$**  (“abstract machine”) if there are functions  $execute : \mathcal{A} \rightarrow Mach$  and  $encode : Sem \rightarrow Mach$  such that the following diagram commutes:

$$\begin{array}{ccc}
 T_{\Sigma(G)} & \xrightarrow{\text{fold}^A} & A \\
 \downarrow \text{fold}^{Sem} & (1) & \downarrow \text{evaluate} \\
 Sem & \xrightarrow[\text{encode}]{} & Mach
 \end{array}$$

*evaluate* runs a “target program”  $a \in A$  on the abstract machine  $Mach$ , while *encode* expresses the source model in terms of the target model.

The initiality of  $T_{\Sigma(G)}$  allows us to reduce the proof that (1) commutes to the extension of *encode* and *evaluate* to  $\Sigma(G)$ -homomorphisms. For this purpose,  $Mach$  must be extended to a  $\Sigma(G)$ -algebra. This can often be done by establishing a target signature  $\Sigma'$  such that  $T_{\Sigma'}$  concides with  $A$ , each constructor of  $\Sigma(G)$  corresponds to a  $\Sigma'$ -term, *Sem* is a  $\Sigma'$ -algebra and *evaluate* folds  $\Sigma'$ -terms in *Sem*. The mapping of  $\Sigma(G)$ -constructors to  $\Sigma'$ -terms may then determine a definition *encode* such that both *encode* and *evaluate* become  $\Sigma(G)$ -homomorphic. In this way, [165] shows the correctness of a compiler that translates imperative programs into flowcharts.

## State unfolding (“operational semantics”)

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature,  $C \in Set^S$  and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .

A **coloring of  $A$  by  $C$**  is an  $S$ -sorted function  $g : A \rightarrow C$ . From now on,  $C^A$  denotes the set of colorings of  $A$  by  $C$ .

Given  $g \in C^A$ , the **coextension  $g^\# : A \rightarrow DT_\Sigma(C)$  of  $g$  to  $DT_\Sigma(C)$** , also called **unfolding**, is the  $\mathcal{T}_p(S, \mathcal{I})$ -sorted function whose values are the labelled trees that are defined as follows:

- For all  $s \in S$  and  $a \in A_s$ ,  $g_s^\#(a)(\epsilon) = g_s(a)$ . (1)

- For all  $s \subseteq \mathcal{I}$  and  $a \in s$ ,  $g_s^\#(a)(\epsilon) = a$ . (2)

- For all  $d : s \rightarrow e \in D$ ,  $a \in A_s$  and  $w \in (\mathcal{I} \cup D)^*$ ,

$$g_s^\#(a)(dw) = g_e^\#(d^{\mathcal{A}}(a))(w). \quad (3)$$

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $a \in A_{e_i}$ ,  $g_e^\#(\iota_i(a)) = i(g_{e_i}^\#(a))$ . (4)

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $a \in A_e$ ,  $i \in I$  and  $w \in (\mathcal{I} \cup D)^*$ ,

$$g_e^\#(a)(\epsilon) = () \quad \text{and} \quad g_e^\#(a)(iw) = g_{e_i}^\#(\pi_i(a))(w). \quad (5)$$

(2), (4) and (5) follow from the lifting of  $g^\#$  as an  $S$ -sorted function to a  $\mathcal{T}_p(S, \mathcal{I})$ -sorted function (see chapter 7).

This is a mutually inductive definition of the elements of the tuple

$$(g^\#(a) : (\mathcal{I} \cup D)^* \rightarrow (\mathcal{I} \cup C) + 1)_{a \in A}$$

of functions. By using the record notation for labelled trees (see chapter 2), (1)  $\wedge$  (3) as well as (5) can be formulated as single equations:

$$g_s^\#(a) = g_s(a)\{d \rightarrow g_e^\#(d^A(a)) \mid d : s \rightarrow e \in D\}. \quad (1) \wedge (3)$$

$$g_s^\#(a) = ()\{i \rightarrow g_{e_i}^\#(\pi_i(a)) \mid i \in I\}. \quad (5)$$

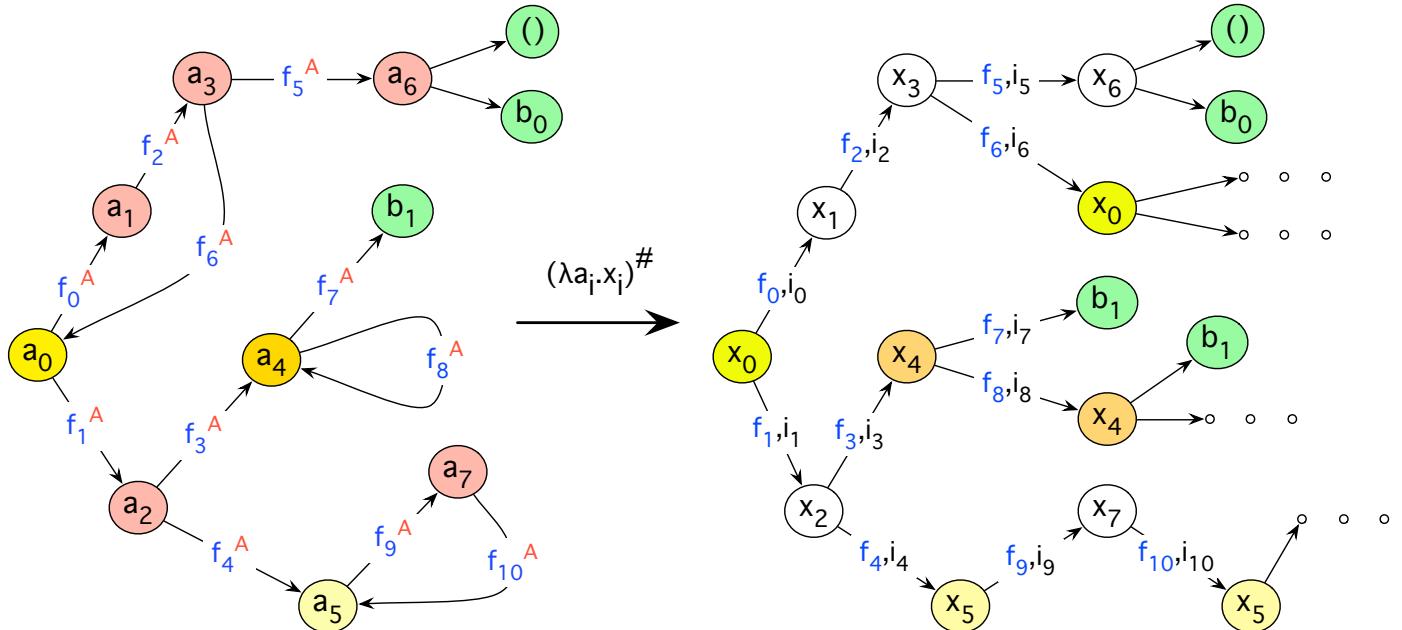


Illustration of unfolding.  $x \xrightarrow{f,i} y$  stands for  $x \xrightarrow{f} i \xrightarrow{\epsilon} y$ .

Intuitively,  $g^\#$  unfolds each “state”  $a \in A$  into the  $\Sigma$ -coterm that represents the “behavior” of  $a$  in  $\mathcal{A}$ .

In particular,  $\text{id}_A^\# : \mathcal{A} \rightarrow DT_\Sigma(A)$  computes for each  $a \in A$  the (tree unfolding of the) transition subgraph of  $\mathcal{A}$  with root  $a$ . For all  $d : s \rightarrow e \in D$  and  $a \in A_s$ ,  $\text{id}_A^\#(a)(d) = d^{\mathcal{A}}(a)$ .

## Theorem DESINV

Let  $a \in A$ ,  $w \in (\mathcal{I} \cup D)^*$ ,  $b = \text{id}_A^\#(a)(w)$  and  $d : s \rightarrow e \in D$ .

(1) If  $b \in A_s$ , then

$$\text{id}_A^\#(a)(wd) = d^{\mathcal{A}}(b).$$

(2) If  $b \in A_e$  for some  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ , then there are  $i \in I$  and  $c \in A_{e_i}$  such that

$$b = \iota_i(c) \quad \text{and} \quad \text{id}_A^\#(a)(w()) = c.$$

(3) If  $b \in A_e$  for some  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ , then for all  $i \in I$ ,

$$\text{id}_A^\#(a)(wi) = \pi_i(b).$$

(4)  $P(a) =_{\text{def}} \text{img}(\text{id}_A^\#(a))$  is the least  $\Sigma$ -invariant  $\langle a \rangle$  of  $\mathcal{A}$  that contains  $a$  (see Products, sums, invariants and quotients of  $\Sigma$ -algebras).

*Proof of (1) by induction on  $|w|$ .* If  $w = \epsilon$ , then  $b = id_A^\#(a)(w) = id^A(a) = a$  and thus

$$id_A^\#(a)(wd) = id_A^\#(a)(d) = d^A(a) = d^A(b).$$

Otherwise  $w = xv$  for some  $x \in \mathcal{I} \cup D$  and  $v \in (\mathcal{I} \cup D)^*$ .

*Case 1:*  $a \in A_s$  for some  $s \in S$ . Then  $x \in D$  and thus

$$id_A^\#(a)(wd) = id_A^\#(x^A(a))(vd) \stackrel{\text{ind. hyp.}}{=} d^A(id_A^\#(x^A(a))(v)) = d^A(id_A^\#(a)(w)) = d^A(b).$$

*Case 2:*  $a = \iota_i(c) \in A_e$  for some  $i \in I$ ,  $c \in A_{e_i}$  and  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = ()$  and thus

$$\begin{aligned} id_A^\#(a)(wd) &= i(id_A^\#(c))(wd) = i(id_A^\#(c))((())vd) = id_A^\#(c)(vd) \stackrel{\text{ind. hyp.}}{=} d^A(id_A^\#(c)(v)) \\ &= d^A(i(id_A^\#(c))((())v)) = d^A(i(id_A^\#(c))(w)) = d^A(id_A^\#(\iota_i(c))(w)) = d^A(id_A^\#(a)(w)) = d^A(b). \end{aligned}$$

*Case 3:*  $a \in A_e$  for some  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = i$  for some  $i \in I$  and thus

$$\begin{aligned} id_A^\#(a)(wd) &= id_A^\#(a)(ivd) = id_A^\#(\pi_i(a))(vd) \stackrel{\text{ind. hyp.}}{=} d^A(id_A^\#(\pi_i(a))(v)) = d^A(id_A^\#(a)(iv)) \\ &= d^A(id_A^\#(a)(w)) = d^A(b). \end{aligned}$$

*Proof of (2) by induction on  $|w|$ .* If  $w = \epsilon$ , then  $a = id_A(a) = b = \iota_i(c)$  for some  $i \in I$  and  $c \in A_{e_i}$  and thus

$$id_A^\#(a)(w()) = id_A^\#(\iota_i(c))() = i(id_A^\#(c))() = id_A^\#(c)(\epsilon) = id_A(c) = c.$$

Otherwise  $w = xv$  for some  $x \in \mathcal{I} \cup D$  and  $v \in (\mathcal{I} \cup D)^*$ .

*Case 1:*  $a \in A_s$  for some  $s \in S$ . Then  $x \in D$  and by induction hypothesis, there are  $i \in I$  and  $c \in A_{e_i}$  such that  $id_A^\#(x^A(a))(v) = \iota_i(c)$  and  $id_A^\#(x^A(a))(v()) = c$ . Therefore,

$$b = id_A^\#(a)(w) = id_A^\#(x^A(a))(v) = \iota_i(c) \text{ and } id_A^\#(a)(w()) = id_A^\#(x^A(a))(v()) = c.$$

*Case 2:*  $a = \iota_j(c') \in A_{e'}$  for some  $j \in J$ ,  $c' \in A_{e'_j}$  and  $e' = \coprod_{j \in J} e_j \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = ()$  and by induction hypothesis, there are  $i \in I$  and  $c \in A_{e_i}$  such that  $id_A^\#(c')(v) = \iota_i(c)$  and  $id_A^\#(c')(v()) = c$ . Therefore,

$$b = id_A^\#(a)(w) = id_A^\#(\iota_j(c'))(w) = j(id_A^\#(c'))(()v) = id_A^\#(c')(v) = \iota_i(c)$$

and

$$id_A^\#(a)(w()) = id_A^\#(\iota_j(c'))(()v()) = j(id_A^\#(c'))(()v()) = id_A^\#(c')(v()) = c.$$

*Case 3:*  $a \in A_{e'}$  for some  $e' = \prod_{j \in J} e'_j \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = j$  for some  $j \in J$ , and by induction hypothesis, there are  $i \in I$  and  $c \in A_{e_i}$  such that  $id_A^\#(\pi_j(a))(v) = \iota_i(c)$  and  $id_A^\#(\pi_j(a))(v()) = c$ . Therefore,

$$b = id_A^\#(a)(w) = id_A^\#(a)(jv) = id_A^\#(\pi_j(a))(v) = \iota_i(c)$$

and

$$id_A^\#(a)(w()) = id_A^\#(a)(jv()) = id_A^\#(\pi_j(a))(v()) = c.$$

*Proof of (3) by induction on  $|w|$ .* If  $w = \epsilon$ , then

$$\begin{aligned} id_A^\#(a)(wi) &= id_A^\#(a)(iw) = id_A^\#(\pi_i(a))(w) = id_A(\pi_i(a)) = \pi_i(a) = \pi_i(id_A(a)) \\ &= \pi_i(id_A^\#(a)(w)) = \pi_i(b). \end{aligned}$$

Otherwise  $w = xv$  for some  $x \in \mathcal{I} \cup D$  and  $v \in (\mathcal{I} \cup D)^*$ .

*Case 1:*  $a \in A_s$  for some  $s \in S$ . Then  $x \in D$  and thus

$$id_A^\#(a)(wi) = id_A^\#(x^A(a))(vi) \stackrel{\text{ind. hyp.}}{=} \pi_i(id_A^\#(x^A(a))(v)) = \pi_i(id_A^\#(a)(w)) = \pi_i(b).$$

*Case 2:*  $a = \iota_j(c) \in A_{e'}$  for some  $j \in J$ ,  $c \in A_{e'_j}$  and  $e' = \coprod_{j \in J} e'_j \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = ()$  and thus

$$\begin{aligned} id_A^\#(a)(wi) &= id_A^\#(\iota_j(c))(wi) = j(id_A^\#(c))((vi)) = id_A^\#(c)(vi) \stackrel{\text{ind. hyp.}}{=} \pi_i(id_A^\#(c)(v)) \\ &= \pi_i(j(id_A^\#(c))((v))) = \pi_i(id_A^\#(\iota_j(c))((v))) = \pi_i(id_A^\#(a)(w)) = \pi_i(b). \end{aligned}$$

*Case 3:*  $a \in A_{e'}$  for some  $e' = \prod_{j \in J} e'_j \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = j$  for some  $j \in J$  and thus

$$\begin{aligned} id_A^\#(a)(wi) &= id_A^\#(a)(jvi) = id_A^\#(\pi_j(a))(vi) \stackrel{\text{ind. hyp.}}{=} \pi_i(id_A^\#(\pi_j(a))(v)) \\ &= \pi_i(id_A^\#(a)(jv)) = \pi_i(id_A^\#(a)(w)) = \pi_i(b). \end{aligned}$$

*Proof of (4).*

Since  $id_A^\#(a)(\epsilon) = id_A(a) = a$ ,  $P(a)$  contains  $a$ .

By (1), for all  $s \in S$ ,  $d : s \rightarrow e \in D$  and  $b \in P(a)_s$ ,  $d^A(b) \in P(a)_e$ . Hence  $P(a)$  is a  $\Sigma$ -invariant of  $\mathcal{A}$ .

Let  $Q(a)$  be a  $\Sigma$ -invariant of  $\mathcal{A}$  that contains  $a$  and  $b \in P(a)$ . Then  $b = id_A^\#(a)(w)$  for some  $w \in (\mathcal{I} \cup D)^*$ .

If  $w = \epsilon$ , then  $b = id_A^\#(a)(\epsilon) = id_A(a) = a \in Q(a)$ .

Otherwise  $w = vx$  for some  $v \in (\mathcal{I} \cup D)^*$  and  $x \in \mathcal{I} \cup D$ . Since  $id_A^\#(a)(v) \in P(a)$ , the induction hypothesis implies  $b' =_{def} id_A^\#(a)(v) \in Q(a)$ .

*Case 1:*  $b' \in A_s$  for some  $s \in S$ . Then  $x \in D$  and thus

$$b = id_A^\#(a)(w) = id_A^\#(a)(vx) = x^A(d_A^\#(a)(v)) = x^A(b') \in Q(a)$$

because  $b' \in Q(a)$  and  $Q(a)$  is a  $\Sigma$ -invariant.

*Case 2:*  $b' \in A_e$  for some  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = ()$  and thus by (2), there are  $i \in I$  and  $c \in A_{e_i}$  such that  $b' = \iota_i(c)$  and  $id_A^\#(a)(v()) = c$ .  $\iota_i(c) = b' \in Q(a)$  implies  $c \in Q(a)$  and thus

$$b = id_A^\#(a)(w) = id_A^\#(a)(v()) = c \in Q(a).$$

Case 3:  $id_A^\#(a)(v) \in A_e$  for some  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = i$  for some  $i \in I$  and thus  $b' \in Q(a)$  implies

$$b = id_A^\#(a)(w) = id_A^\#(a)(vi) \stackrel{(3)}{=} \pi_i(id_A^\#(a)(v)) = \pi_i(b') \in Q(a).$$

Hence  $b \in Q(a)$  in all three cases, and we conclude that  $P(a)$  is the *least* invariant of  $\mathcal{A}$  that contains  $a$ .  $\square$

Let  $\mathcal{B}$  be a  $\Sigma$ -algebra with carrier  $B$ . For all  $\Sigma$ -homomorphisms  $h : \mathcal{A} \rightarrow \mathcal{B}$ ,

$$h(\langle a \rangle) = \langle h(a) \rangle. \quad (5)$$

*Proof.* For all  $w \in (\mathcal{I} \cup D)^*$ ,  $h(id_A^\#(a)(w)) = h^{(\mathcal{I} \cup D)^*}(id_A^\#(a))(w) = id_B^\#(h(a))(w)$ . Hence both  $h(\langle a \rangle) \subseteq \langle h(a) \rangle$  and  $\langle h(a) \rangle \subseteq h(\langle a \rangle)$ .  $\square$

If  $\mathcal{A} = DT_\Sigma(C)$ , then  $g$  is called a **recoloring** because  $g^\# : DT_\Sigma(C) \rightarrow DT_\Sigma(C)$  only changes the labels of the internal nodes of coterms: For all coterms  $t$  and subcoterms  $u$  of  $t$ ,  $g^\#$  labels the root of  $u$  with  $g(u)$ .

For all  $t \in DT_\Sigma(C)$  and  $v \in (\mathcal{I} \cup D)^*$ ,  $\lambda w.t(vw) = id_{DT_\Sigma(C)}^\#(t)(v)$ .  $(6)$

Proof by induction on  $|v|$ . Let  $t' = \lambda w.t(vw)$  and  $\textcolor{red}{h} = id_{DT_\Sigma(C)}^\#(t)$ .

If  $v = \epsilon$ , then

$$t' = t = id_{DT_\Sigma(C)}(t) = h(\epsilon) = h(v).$$

Otherwise  $\textcolor{blue}{v} = v'\textcolor{blue}{x}$  for some  $v' \in (\mathcal{I} \cup D)^*$  and  $x \in \mathcal{I} \cup D$ . Hence \*\*\*\*

$$t' = \lambda w.t(vw) = \lambda w.t(v'xw) = \lambda w.(\lambda w'.t(v'w'))(xw) \stackrel{\text{ind. hyp.}}{=} \lambda w.h(v')(xw). \quad (7)$$

Case 1.  $h(v') \in DT_\Sigma(C)_s$  for some  $s \in S$ . Then  $x \in D$  and thus

$$t' \stackrel{(7)}{=} \lambda w.h(v')(x \cdot w) = \lambda w.x^{DT_\Sigma(C)}(h(v'))(w) = x^{DT_\Sigma(C)}(h(v')) \stackrel{(1)}{=} h(v' \cdot x) = h(v).$$

Case 2.  $h(v') \in DT_\Sigma(C)_e$  for some  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = ()$  and thus by (2), there are  $i \in I$  and  $u \in DT_\Sigma(C)_{e_i}$  such that  $h(v') = i(u)$  and  $h(v'()) = u$ . Hence

$$t' \stackrel{(7)}{=} \lambda w.h(v')(x \cdot w) = \lambda w.i(u)((()w)) = \lambda w.u(w) = u = h(v'()) = h(v'x) = h(v).$$

Case 3.  $h(v') \in DT_\Sigma(C)_e$  for some  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then  $x = i$  for some  $i \in I$  and thus

$$t' \stackrel{(7)}{=} \lambda w.h(v')(x \cdot w) = \lambda w.\pi_i(h(v'))(w) = \pi_i(h(v')) \stackrel{(3)}{=} h(v'i) = h(v'x) = h(v). \quad \square$$

By (6), for all  $t \in DT_\Sigma(C)$ ,  $\langle t \rangle$  is the set of subtrees of  $t$ .

Given  $t \in DT_{\Sigma}(C)$ ,  $g \in C^A$  solves the **coequation**  $ex(t)$  in  $\mathcal{A}$ , written as  $\mathcal{A} \models_g ex(t)$ , if  $g^{\#}(a) = t$  for some  $a \in A$ .

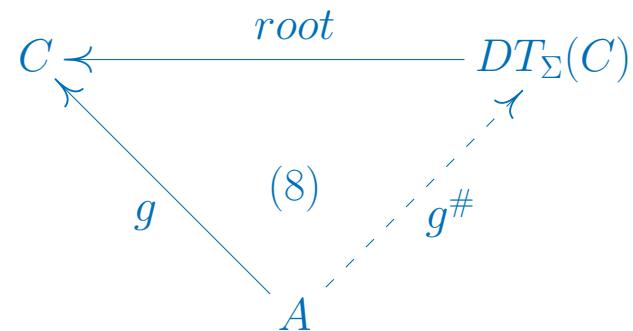
$\mathcal{A}$  satisfies a **conditional coequation**  $ex(t) \Rightarrow \bigvee_{i=1}^n ex(t_i)$  if every  $g \in C^A$  that solves  $ex(t)$  also solves  $ex(t_i)$  for some  $1 \leq i \leq n$ .

An empty disjunction is identified with *False*.

Consequently,  $\mathcal{A}$  satisfies  $ex(t) \Rightarrow False$  if for all  $g \in C^A$  and  $a \in A$ ,  $g^{\#}(a) \neq t$ .

## Theorem COFREE

Let  $C \in Set^S$ . Then  $DT_{\Sigma}(C)$  is a **cofree  $\Sigma$ -algebra over  $C$** , i.e., for all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$  and  $g \in C^A$ ,  $g^{\#}$  is the only  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $DT_{\Sigma}(C)$  that satisfies (8).



In particular, if for all  $s \in S$ ,  $C_s = 1$ , then there is exactly one coloring  $g \in C^A$ , (11) reduces to the uniqueness of  $g^*$  and thus  $DT_\Sigma = DT_\Sigma(C)$  is final in  $Alg_\Sigma$ .  $g^\#$  no longer depends on  $g$  and is denoted by  $unfold^{\mathcal{A}}$ .

*Proof.* By (1),  $g^\#$  satisfies (8).

$g^\#$  is  $\Sigma$ -homomorphic: For all  $a \in A_s$  and  $d : s \rightarrow e \in D$ ,

$$d^{DT_\Sigma(C)}(g_s^\#(a)) = d^{DT_\Sigma(C)}(g_s^\#(a)\{d \rightarrow g_s^\#(d^{\mathcal{A}}(a)) \mid d : s \rightarrow e \in D\}) = g_s^\#(d^{\mathcal{A}}(a)).$$

$g^\#$  is unique: Let  $h : \mathcal{A} \rightarrow DT_\Sigma(C)$  be a  $\Sigma$ -homomorphism with  $root \circ h = g$ .

- For all  $s \in S$  and  $a \in A_s$ ,  $g_s^\#(a)(\epsilon) = g_s(a) \stackrel{root \circ h = g}{=} h_s(a)(\epsilon)$ .
- For all  $d : s \rightarrow e \in D$ ,  $a \in A_s$  and  $w \in (\mathcal{I} \cup D)^*$ ,

$$\begin{aligned} g_s^\#(a)(dw) &= g_e^\#(d^{\mathcal{A}}(a))(w) \stackrel{ind. \ hyp.}{=} h_e(d^{\mathcal{A}}(a))(w) \stackrel{h \ hom.}{=} d^{DT_\Sigma(C)}(h_s(a))(w) \\ &= h_s(a)(dw). \end{aligned}$$

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $a \in A_{e_i}$ ,

$$g_e^\#(\iota_i(a)) = i(g_{e_i}^\#(a)) \stackrel{ind. \ hyp.}{=} i(h_{e_i}(a)) = \iota_i(h_{e_i}(a)) = h_e(\iota_i(a)).$$

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $a \in \prod_{i \in I} A_{e_i}$ ,

$$\pi_i(g_e^\#(a)) = \pi_i((\{i \rightarrow g_{e_i}^\#(\pi_i(a)) \mid i \in I\}) = g_{e_i}^\#(\pi_i(a)) \stackrel{\text{ind. hyp.}}{=} h_{e_i}(\pi_i(a)) = \pi_i(h_e(a)).$$

Hence for all  $e \in \mathcal{T}_p(S, \mathcal{I})$ ,  $g_e^\# = h_e$ . □

## Lemma RECOL

For all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$ ,  $g \in C^A$  and  $\Sigma$ -homomorphisms  $h : \mathcal{B} \rightarrow \mathcal{A}$ ,

$$(g \circ h)^\# = g^\# \circ h.$$

*Proof.* Since  $\text{root} \circ g^\# \circ h = g \circ h$ , the conjecture follows from the fact that  $(g \circ h)^\#$  is the only  $\Sigma$ -homomorphism  $h' : \mathcal{B} \rightarrow DT_\Sigma(C)$  with  $\text{root} \circ h' = g \circ h$ . □

Since  $g^\#$  is the only  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $DT_\Sigma(C)$  that satisfies (11),  $\text{unfold}^\mathcal{A}$  is the only  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $DT_\Sigma$ , i.e.,  **$DT_\Sigma$  is final in  $\text{Alg}_\Sigma$** .

For all  $a \in A$ ,  $(\mathcal{A}, a)$  realizes  $t \in DT_\Sigma$  if  $t = \text{unfold}^\mathcal{A}(a)$ .

## Corollary DESMIN

For all  $t \in DT_\Sigma$ ,  $(\langle t \rangle, t)$  is a minimal realization of  $t$ .

*Proof.* Since  $DT_\Sigma$  is final in  $Alg_\Sigma$ ,  $unfold^{DT_\Sigma} = id_{DT_\Sigma}$ . Hence

$$t = unfold^{DT_\Sigma}(t) = unfold^{DT_\Sigma}(inc_{\langle t \rangle}(t)) = unfold^{\langle t \rangle}(t)$$

and thus  $(\langle t \rangle, t)$  realizes  $t$ . Moreover, let  $\mathcal{A}$  be a  $\Sigma$ -algebras with carrier  $A$  that realizes  $t$ . Then for some  $a \in A$ ,

$$|\langle t \rangle| = |\langle unfold^{\mathcal{A}}(a) \rangle| \stackrel{(5)}{=} |unfold^{\mathcal{A}}(\langle a \rangle)| \leq |\langle a \rangle|,$$

and thus  $(\langle t \rangle, t)$  is minimal. □

A  $\Sigma$ -algebra  $\mathcal{A}$  is **behaviorally complete** if  $unfold^{\mathcal{A}}$  is epi.

$\mathcal{A}$  is **observable** (or **cogenerated**) if  $unfold^{\mathcal{A}}$  is mono.

$OAlg_\Sigma$  denotes the full subcategory of  $Alg_\Sigma$  whose objects are all observable  $\Sigma$ -algebras.

Since for all  $t \in DT_\Sigma$  and  $t' \in \langle t \rangle$ ,

$$t' = unfold^{DT_\Sigma}(t') = unfold^{DT_\Sigma}(inc_{\langle t \rangle}(t')) = unfold^{\langle t \rangle}(t'),$$

$\langle t \rangle$  is observable.

By Lemma IMG (2), for all  $\mathcal{A} \in OAlg_\Sigma$ ,  $\mathcal{A} \cong DT_\Sigma|_{img(unfold^{\mathcal{A}})}$ .

## Lemma FINOBS

Let  $\mathcal{K}$  be a full subcategory of  $OAlg_{\Sigma}$  and the  $\Sigma$ -algebra  $\mathcal{B} = \mathcal{B}(\mathcal{K})$  be defined as follows:

- For all  $s \in S$ ,  $B_s = img([unfold^{\mathcal{A}}]_{\mathcal{A} \in \mathcal{K}})_s = \bigcup_{\mathcal{A} \in \mathcal{K}} img(unfold^{\mathcal{A}})_s$  (see sum equation 20) and  $\mathcal{B}(s) = DT_{\Sigma,s}|_{B_s}$ .
- For all  $d : s \rightarrow e \in D$  and  $t \in DT_{\Sigma}$ ,  $d^{\mathcal{B}}(t) = d^{DT_{\Sigma}}(t)$ .

For all  $\mathcal{A} \in \mathcal{K}$ , there is a unique  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $\mathcal{B}$ .

In particular,  $\mathcal{B}$  is final in  $OAlg_{\Sigma}$ .

*Proof.* By Lemma IMG (2), there is a unique  $\Sigma$ -epimorphism  $h : \coprod \mathcal{K} \rightarrow \mathcal{B}$  such that  $inc_{\mathcal{B}} \circ h = [unfold^{\mathcal{A}}]_{\mathcal{A} \in \mathcal{K}}$ . Hence for all  $\mathcal{A} \in \mathcal{K}$ ,  $h \circ \iota_{\mathcal{A}} : \mathcal{A} \rightarrow \mathcal{B}$  is  $\Sigma$ -homomorphic. Suppose that there are two  $\Sigma$ -homomorphisms  $h_1, h_2 : \mathcal{A} \rightarrow \mathcal{B}$ . Since  $DT_{\Sigma}$  is final in  $Alg_{\Sigma}$ ,  $inc_{\mathcal{B}} \circ h_1 = inc_{\mathcal{B}} \circ h_2$ . Hence  $h_1 = h_2$  because  $inc_{\mathcal{B}}$  is mono.

In particular, since  $\mathcal{B} \in OAlg_{\Sigma}$ ,  $\mathcal{B}$  is final in  $OAlg_{\Sigma}$ . □

## Example

Let  $\Sigma = Acc(X)$ ,  $\mathcal{C}$  be a  $Med(X)$ -algebra and  $\mathcal{K}$  be the category of observable  $\Sigma$ -algebras  $\mathcal{A}$  with  $\mathcal{A}|_{Med(X)} = \mathcal{C}$ . Then  $\mathcal{B}(\mathcal{K})$  agrees with the cofree (2-)pointed automaton over  $\mathcal{C}$  as defined in [149], section 5, and embedded in the final observable  $\Sigma$ -algebra. □

## Coterm grounding

Let  $\mathcal{I}(C) = \mathcal{I} \cup C$  and  $\mathcal{D}(C) = D \cup \{col_s : s \rightarrow C_s \mid s \in S\}$ . Then

$$\Sigma(C) = (S, \mathcal{I}(C), \mathcal{D}(C))$$

is called the **grounding of  $\Sigma$  on  $C$** .

$DT_\Sigma(C)$  is a  $\Sigma(C)$ -algebra: For all  $s \in S$  and  $t \in DT_\Sigma(C)$ ,  $col_s^{DT_\Sigma(C)}(t) =_{def} t(\epsilon)$ .

Let  $\mathcal{A}$  be a  $\Sigma(C)$ -algebra with carrier  $A$ . Since

$$col^{DT_\Sigma(C)} \circ (col^\mathcal{A})^\# = root \circ (col^\mathcal{A})^\# = col^\mathcal{A},$$

$(col^\mathcal{A})^\#$  is compatible with  $col$  and thus  $\Sigma(C)$ -homomorphic. Vice versa, two  $\Sigma(C)$ -homomorphisms  $h, h' : \mathcal{A} \rightarrow DT_\Sigma(C)$  are compatible with  $col$ . Hence

$$root \circ h = col^{DT_\Sigma(C)} \circ h = col^\mathcal{A} = col^{DT_\Sigma(C)} \circ h' = root \circ h'.$$

Since  $h$  and  $h'$  are  $\Sigma$ -homomorphic, we conclude  $h = h'$ .

Hence  $DT_\Sigma(C)$  is final in  $Alg_{\Sigma(C)}$  and for all  $\mathcal{A} \in Alg_{\Sigma(C)}$ ,

$$unfold^\mathcal{A} = (col^\mathcal{A})^\#.$$

By replacing the label  $c \in C$  of every inner node  $n$  of  $t \in DT_\Sigma(C)$  with  $\epsilon$  and adding an edge to  $t$  with source  $n$ , label  $col$  and a new target node labelled with  $x$ , we obtain a  $\Sigma(C)$ -isomorphism from  $DT_\Sigma(C)$  to  $DT_{\Sigma(C)}$ .

Intuitively, the  $\Sigma$ -isomorphism  $DT_{\Sigma(C)}|_{\Sigma} \cong DT_{\Sigma}(C)$  is obtained by replacing each edge  $e$  of  $t$  labelled with  $col$  with its source node and labelling this node with the target of  $e$ . Moreover,  $H_{\Sigma(C)} = H_{\Sigma} \times C$  (see  $\Sigma$ -functors).

## Sample final algebras

Since final algebras are unique up to isomorphism,

- $DT_{coNat} \cong \mathbb{N}_{\infty}$  is final in  $Alg_{coNat}$  (see sample algebra 2), (1)

- $DT_{Stream(X)} \cong InfSeq(X)$  is final in  $Alg_{Stream}$  (see sample algebra 5), (2)

- $DT_{coDyn(X,Y)} \cong coSeq(X, Y)$  is final in  $Alg_{coDyn(X,Y)}$  (see sample algebra 8), (3)

- $DT_{coNelist(X)} \cong Neseq(X)$  is final in  $Alg_{coNelist(X)}$  (see sample algebra 9), (4)

- $DT_{infBintree(X)} \cong InfBin(X)$  is final in  $Alg_{infBintree(X)}$  (see sample algebra 11),

- $DT_{coBintree(X)} \cong Bin(X)$  is final in  $Alg_{coBintree(X)}$  (see sample algebra 12),

- $DT_{infTree(X)} \cong FBInfTree(X)$  is final in  $Alg_{infTree(X)}$  (see sample algebra 16),

- $DT_{coTree_{\omega}(X)} \cong FBTree(X)$  is final in  $Alg_{coTree_{\omega}(X)}$  (see sample algebra 17),

- $DT_{coTree(X)} \cong Tree_{\infty}(X)$  is final in  $Alg_{coTree(X)}$  (see sample algebra 18),

- $DT_{Med(X)} \cong 1$  is final in  $Alg_{Med(X)}$ ,

- $DT_{NMed^*(X)} \cong otr(X \times \mathbb{N}, (\Delta_X, <), 1)$  is final in  $Alg_{NMed^*(X)}$ ,
- $DT_{DAut(X,Y)} \cong Beh(X, Y)$  is final in  $Alg_{DAut(X,Y)}$  (see sample algebra 24), (5)
- $DT_{Acc(X)} \cong Pow(X)$  is final in  $Alg_{Acc(X)}$  (see sample algebra 20), (6)
- $DT_{NAcc(X)} \cong NPow(X)$  is final in  $Alg_{NAcc(X)}$  (see sample algebra 21), (7)
- $DT_{Mealy(X,Y)} \cong MBeh(X, Y) \cong Causal(X, Y)$  is final in  $Alg_{Mealy(X,Y)}$  (see sample algebra 26), (8)
- $DT_{PAut(X,Y)} \cong PBeh(X, Y)$  is final in  $Alg_{PAut(X,Y)}$  (see sample algebra 27), (9)
- $DT_{NAut^*(X,Y)} \cong NBeh(X, Y)$  is final in  $Alg_{NAut^*(X,Y)}$  (see sample algebra 28), (10)
- $DT_{TAcc(\Sigma)} \cong TPow(\Sigma)$  is final in  $Alg_{TAcc(\Sigma)}$  (see sample algebra 29), (11)
- $DT_{NTAcc(\Sigma)} \cong NTPow(\Sigma)$  is final in  $Alg_{NTAcc(\Sigma)}$  (see sample algebra 30), (12)
- $DT_{NTAcc^*(\Sigma)} \cong NTPow^*(\Sigma)$  is final in  $Alg_{NTAcc^*(\Sigma)}$  (see sample algebra 31). (13)

By (5) and since  $DT_{DAut(X,Y)}|_{Med(X)}$  and  $DT_{Med(X)}(Y)$  are  $Med(X)$ -isomorphic,  $Beh(X, Y)|_{Med(X)}$  is cofree over  $Y$  in  $Alg_{Med(X)}$  (see also section 19.4).

Given a destructive signature  $\Sigma$  and a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$ ,  $\text{unfold}^{\mathcal{A}}$  denotes the unique  $\Sigma$ -homomorphism from  $\mathcal{A}$  not only to  $DT_{\Sigma}$ , but also to isomorphic representations of  $DT_{\Sigma}$  like those listed above.

In cases (1)-(13), the respective definition of  $h =_{def} \text{unfold}^{\mathcal{A}}$  reads as follows:

### 1. $coNat$ -algebra $\mathbb{N}_{\infty}$

$$\begin{aligned} h : A &\rightarrow \mathbb{N}_{\infty} \\ a &\mapsto \max\{n \in \mathbb{N}_{\infty} \mid \forall 1 \leq i < n : (\text{pred}^{\mathcal{A}})^i(a) \neq ()\} \end{aligned}$$

where  $\max$  denotes the maximum w.r.t. the usual (well-founded) ordering on  $\mathbb{N}_{\infty}$ , including  $n < \infty$  for all  $n \in \mathbb{N}$  (see [6], Examples 3.8).

$h$  is  $coNat$ -homomorphic:

Case 1:  $\text{pred}^{\mathcal{A}}(a) = \epsilon$ . Then  $h(a) = 0$ . Hence

$$\text{pred}^{\mathbb{N}_{\infty}}(h(a)) = \text{pred}^{\mathbb{N}_{\infty}}(0) = \epsilon = \text{pred}^{\mathcal{A}}(a).$$

Case 2:  $h(\text{pred}^{\mathcal{A}}(a)) \in \mathbb{N}$ . Then  $h(a) = h(\text{pred}^{\mathcal{A}}(a)) + 1$ . Hence

$$\text{pred}^{\mathbb{N}_{\infty}}(h(a)) = \text{pred}^{\mathbb{N}_{\infty}}(h(\text{pred}^{\mathcal{A}}(a)) + 1) = h(\text{pred}^{\mathcal{A}}(a)).$$

Case 3:  $h(pred^{\mathcal{A}}(a)) = \omega$ . Then  $h(a) = \omega$ . Hence

$$pred^{\mathbb{N}_\infty}(h(a)) = pred^{\mathbb{N}_\infty}(\omega) = \omega = h(pred^{\mathcal{A}}(a)).$$

$h$  is the only *coNat*-homomorphism from  $\mathcal{A}$  to  $\mathbb{N}_\infty$ : Let  $h' : \mathcal{A} \rightarrow \mathbb{N}_\infty$  be *coNat*-homomorphic.

Case 1:  $h'(a) = 0$ . Then

$$\epsilon = pred^{\mathbb{N}_\infty}(h'(a)) \stackrel{h' \text{ hom.}}{=} pred^{\mathcal{A}}(a)$$

and thus  $h(a) = 0 = h'(a)$ .

Case 2:  $0 < h'(a) \in \mathbb{N}$ . Then  $pred^{\mathbb{N}_\infty}(h'(a)) = h'(a) - 1 \neq ()$ . Hence

$$h(pred^{\mathcal{A}}(a)) \stackrel{\text{ind. hyp.}}{=} h'(pred^{\mathcal{A}}(a)) \stackrel{h' \text{ hom.}}{=} pred^{\mathbb{N}_\infty}(h'(a)) = h'(a) - 1$$

and thus  $h'(a) = h(pred^{\mathcal{A}}(a)) + 1 = h(a)$ . The induction hypothesis is based on the well-founded ordering  $\gg$  on  $A$  with  $a \gg b \Leftrightarrow_{\text{def}} h(a) > h(b)$ . Note that for all  $a \in A$ ,  $pred^{\mathcal{A}}(a) \neq ()$  implies  $a \gg pred^{\mathcal{A}}(a)$ .

Case 3:  $h'(a) = \omega$ . Then

$$\infty = pred^{\mathbb{N}_\infty}(\infty) = pred^{\mathbb{N}_\infty}(h'(a))) \stackrel{h' \text{ hom.}}{=} h'(pred^{\mathcal{A}}(a)) \stackrel{\text{ind. hyp.}}{=} h(pred^{\mathcal{A}}(a)).$$

Hence  $h(a) = \omega = h'(a)$ .

Exercise 16 Define a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A = \mathbb{N}_\infty^2$  such that for all  $m, n \in \mathbb{N}_\infty$ ,

$$+\infty : \mathbb{N}_\infty^2 \rightarrow \mathbb{N}_\infty$$

$$(m, n) \mapsto \text{if } m, n \in \mathbb{N} \text{ then } m + n \text{ else } \infty$$

agrees with  $\text{unfold}^{\mathcal{A}}$ .

## 2. $\text{Stream}(X)$ -algebra $\text{InfSeq}(X)$

$$h : A \rightarrow X^\mathbb{N}$$

$$a \mapsto \lambda n. \text{head}^{\mathcal{A}}((\text{tail}^{\mathcal{A}})^n(a))$$

## 3. $\text{coDyn}(X, Y)$ -algebra $\text{coSeq}(X, Y)$

$$h : A \rightarrow X^* \times Y \cup X^\mathbb{N}$$

$$a \mapsto \begin{cases} (xw, y) & \text{if } \text{split}^{\mathcal{A}}(a) = \iota_1(x, b) \wedge h(b) = (w, y) \in X^* \times Y \\ \lambda n. \text{if } n = 0 \text{ then } x \\ \quad \text{else } h(b)(n - 1) & \text{if } \text{split}^{\mathcal{A}}(a) = \iota_1(x, b) \wedge h(b) \in X^\mathbb{N} \\ (\epsilon, y) & \text{if } \text{split}^{\mathcal{A}}(a) = \iota_2(y) \end{cases}$$

Analogously to the definition of unfolding into coterms (see [State unfolding](#)), this is a mutually inductive definition of the elements of the tuple  $(h(a))_{a \in A}$  of functions.

#### 4. $coNelist(X)$ -algebra $Neseq(X)$

$$h : A \rightarrow X^+ \cup X^{\mathbb{N}}$$

$$a \mapsto \begin{cases} xw & \text{if } \text{split}^{\mathcal{A}}(a) = \iota_1(x, b) \wedge h(b) = w \in X^* \times Y \\ \lambda n. \text{if } n = 0 \text{ then } x \\ \quad \text{else } h(b)(n - 1) & \text{if } \text{split}^{\mathcal{A}}(a) = \iota_1(x, b) \wedge h(b) \in X^{\mathbb{N}} \\ \epsilon & \text{if } \text{split}^{\mathcal{A}}(a) = \iota_2() \end{cases}$$

This is a mutually inductive definition of the elements of the tuple  $(h(a))_{a \in A}$  of functions.

#### 5. $DAut(X, Y)$ -algebra $Beh(X, Y)$

$$h : A \rightarrow Y^{X^*}$$

$$a \mapsto \lambda w. \text{if } w = \epsilon \text{ then } \beta^{\mathcal{A}}(a) \text{ else } h(\delta^{\mathcal{A}}(a)(\text{head}(w)))(\text{tail}(w))$$

This is a mutually inductive definition of the elements of the tuple  $(h(a) : X^* \rightarrow Y)_{a \in A}$  of functions.

For all  $a \in A$ ,  $(\mathcal{A}, a)$  realizes the behaviour function  $h(a)$ .

$id_A^\# : A \rightarrow A^{X^*}$  is often denoted by  $(\delta^{\mathcal{A}})^*$ .

Let  $Y$  be a semiring. Then  $Beh(X, Y)$  is a linear automaton (see sample algebra 25). Moreover, if  $A$  is a  $Y$ -semimodule, then  $h$  is linear (see [33], Theorem 2). Consequently,  $Beh(X, Y)$  is final in the category  $Alg_\Sigma \cap SMod_R$ .

## 6. $Acc(X)$ -algebra $Pow(X)$

$$h : A \rightarrow \mathcal{P}(X^*)$$

$$a \mapsto \begin{cases} \{x \cdot w \mid x \in X, w \in h(\delta^{\mathcal{A}}(a)(x))\} & \text{if } \beta^{\mathcal{A}}(a) = 0 \\ \{x \cdot w \mid x \in X, w \in h(\delta^{\mathcal{A}}(a)(x))\} \cup 1 & \text{if } \beta^{\mathcal{A}}(a) = 1 \end{cases}$$

This is a mutually inductive definition of the elements of the tuple  $(h(a))_{a \in A}$  of subsets of  $X^*$  where  $h(a) \subseteq X^*$  is regarded as its indicator function  $set2fun(h(a)) : X^* \rightarrow 2$  (see Preliminaries).

For all  $a \in A$ ,  $(\mathcal{A}, a)$  accepts the language  $h(a)$ .

## 7. $N\text{Acc}(X)$ -algebra $NPow(X)$

$$h : A \rightarrow \mathcal{P}(X^*)$$

$$a \mapsto \begin{cases} \{x \cdot w \mid x \in X, \exists b \in \delta^{\mathcal{A}}(a)(x) : w \in h(b)\} & \text{if } \beta^{\mathcal{A}}(a) = 0 \\ \{x \cdot w \mid x \in X, \exists b \in \delta^{\mathcal{A}}(a)(x) : w \in h(b)\} \cup 1 & \text{if } \beta^{\mathcal{A}}(a) = 1 \end{cases}$$

Again, this is a mutually inductive definition of the elements of the tuple  $(h(a))_{a \in A}$  of subsets of  $X^*$ .

## 8. $Mealy(X, Y)$ -algebra $M\text{Beh}(X, Y)$

$$h : A \rightarrow Y^{X^+}$$

$$a \mapsto \lambda w. \text{if } |w| = 1 \text{ then } \beta^{\mathcal{A}}(a)(\text{head}(w)) \text{ else } h(\delta^{\mathcal{A}}(a)(\text{head}(w)))(\text{tail}(w))$$

This is a mutually inductive definition of the elements of the tuple  $(h(a) : X^+ \rightarrow Y)_{a \in A}$  of functions.

$Mealy(X, Y)$ -algebra  $Causal(X, Y)$  (see Preliminaries):

$$h : A \rightarrow \mathcal{C}(X, Y)$$

$$a \mapsto \lambda f \lambda n. \text{if } n = 0 \text{ then } \beta^{\mathcal{A}}(a)(f(0))$$

$$\quad \quad \quad \text{else } h(\delta^{\mathcal{A}}(a)(f(0)))(\lambda k. f(k + 1))(n - 1)$$

This is a mutually inductive definition of the elements of the tuple

$$(h(a)(f) : \mathbb{N} \rightarrow Y)_{a \in A, f : \mathbb{N} \rightarrow X}$$

of functions.

## 9. $PAut(X, Y)$ -algebra $PBeh(X, Y)$

$$\begin{aligned} h : A &\rightarrow ltr(X, Y) \\ a &\mapsto \beta^A(a)\{x \rightarrow h(b) \mid x \in X, \delta^A(a)(x) = b \in A\} \end{aligned}$$

This is a mutually inductive definition of the elements of the tuple  $(h(a) : X^* \rightarrow Y+1)_{a \in A}$  of functions.

## 10. $NAut^*(X, Y)$ -algebra $NBeh(X, Y)$

$$\begin{aligned} h : A &\rightarrow otr(X \times \mathbb{N}, (\Delta_X, <), Y) \\ a &\mapsto \beta^A(a)\{(x, i) \rightarrow h(a_i) \mid x \in X, 1 \leq i \leq n, \delta^A(a)(x) = (a_1, \dots, a_n)\} \end{aligned}$$

Again, this is a mutually inductive definition of the components of the tuple

$$(h(a) : X^* \rightarrow Y + 1)_{a \in A}$$

of partial functions.

Exercise 17 Show that  $h$  is  $\Sigma$ -homomorphic, i.e., for all  $a \in A_{state}$ ,

$$\begin{aligned}\delta^{NBeh(X,Y)}(h(a)) &= map(h) \circ \delta^{\mathcal{A}}(a), \\ \beta^{NBeh(X,Y)}(h(a)) &= \beta^{\mathcal{A}}(a).\end{aligned}$$

Let  $\Sigma = (S, \mathcal{I}, C)$  be a finitary signature.

## 11. $TAcc(\Sigma)$ -algebra $TPow(\Sigma)$

$$\begin{aligned}h : A &\rightarrow \mathcal{P}(T_\Sigma) \\ a &\mapsto \{c(t_1, \dots, t_n) \mid c \in C, \delta_c^{\mathcal{A}}(a) = (a_1, \dots, a_n), \\ &\quad \forall 1 \leq i \leq n : t_i \in h(a_i)\}\end{aligned}$$

This is a mutually inductive definition of the elements of the tuple  $(h(a))_{a \in A}$  of subsets of  $T_\Sigma$  where  $h(a) \subseteq T_\Sigma$  is regarded as its indicator function  $set2fun(h(a)) : T_\Sigma \rightarrow 2$  (see Preliminaries).

## 12. $NTAcc(\Sigma)$ -algebra $NTPow(\Sigma)$

$$h : A \rightarrow \mathcal{P}(T_\Sigma)$$

$$\begin{aligned} a \mapsto \{c(t_1, \dots, t_n) \mid c \in C, \delta_c^A(a) &= \{(a_{ij})_{j=1}^n \mid 1 \leq i \leq k\} \\ &\Rightarrow \forall 1 \leq j \leq n \exists 1 \leq i_j \leq k : t_i \in h(a_{ij})\} \end{aligned}$$

Again, this is a mutually inductive definition of the elements of the tuple  $(h(a))_{a \in A}$  of subsets of  $T_\Sigma$ .

## 13. $NTAcc^*(\Sigma)$ -algebra $NTPow^*(\Sigma)$

$$h : A \rightarrow \mathcal{P}(T_\Sigma)$$

$$\begin{aligned} a \mapsto \{c(t_1, \dots, t_n) \mid c \in C, \delta_c^A(a) &= [(a_{ij})_{j=1}^n \mid 1 \leq i \leq k] \\ &\Rightarrow \forall 1 \leq j \leq n \exists 1 \leq i_j \leq k : t_i \in h(a_{ij})\} \end{aligned}$$

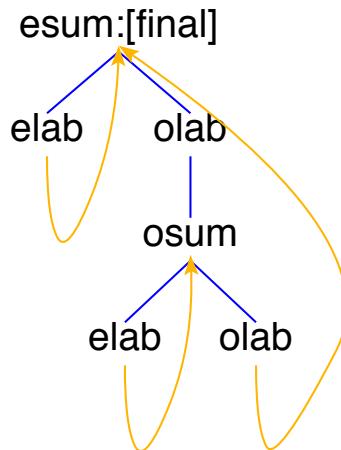
Again, this is a mutually inductive definition of the elements of the tuple  $(h(a))_{a \in A}$  of subsets of  $T_\Sigma$ .

## Example EOSUM (see sample algebras 7 and 20)

Define  $h : eo \rightarrow Pow(\mathbb{Z})$  as follows:

$$\begin{aligned} h(esum) &= \{(x_1, \dots, x_n) \in \mathbb{Z}^* \mid \sum_{i=1}^n x_i \text{ is even}\}, \\ h(osum) &= \{(x_1, \dots, x_n) \in \mathbb{Z}^* \mid \sum_{i=1}^n x_i \text{ is odd}\}. \end{aligned}$$

Transitions and atoms of  $eo$ :



$(eo, esum)$  accepts  $h(esum)$ . (1)

$(eo, osum)$  accepts  $h(osum)$ . (2)

*Proof of (1) and (2).* By Lemma FIN (2) and (3), all  $DAut(\mathbb{Z}, 2)$ -homomorphisms from  $eo$  to  $Pow(\mathbb{Z})$  agree with  $unfold^{eo} : eo \rightarrow Pow(\mathbb{Z})$ .

Hence (1) and (2) hold true if  $h$  is  $DAut(\mathbb{Z}, 2)$ -homomorphic, i.e., if  $h$  satisfies the following equations for all  $st \in \{esum, osum\}$  and  $x \in \mathbb{Z}$ ,

$$h(\delta^{eo}(st)(x)) = \delta^{Pow}(h(st))(x), \quad (3)$$

$$\beta^{eo}(st) = \beta^{Pow}(h(st)). \quad (4)$$

*Proof of (3).* Let  $st = esum$  and  $x$  be even. Then

$$w \in h(\delta^{eo}(esum)(x)) \Leftrightarrow w \in h(esum) \Leftrightarrow x \cdot w \in h(esum) \Leftrightarrow w \in \delta^{Pow}(h(esum))(x).$$

Let  $st = esum$  and  $x$  be odd. Then

$$w \in h(\delta^{eo}(osum)(x)) \Leftrightarrow w \in h(esum) \Leftrightarrow x \cdot w \in h(osum) \Leftrightarrow w \in \delta^{Pow}(h(osum))(x).$$

Let  $st = osum$  and  $x$  be even. Then

$$w \in h(\delta^{eo}(osum)(x)) \Leftrightarrow w \in h(osum) \Leftrightarrow x \cdot w \in h(osum) \Leftrightarrow w \in \delta^{Pow}(osum)(x).$$

Let  $st = osum$  and  $x$  be odd. Then

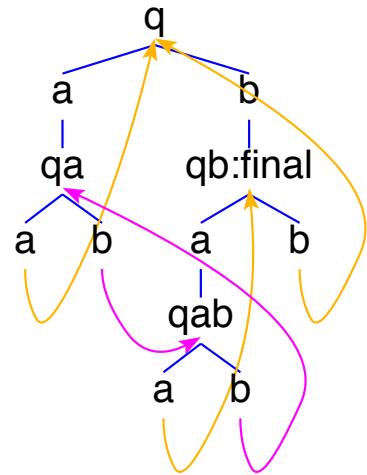
$$w \in h(\delta^{eo}(osum)(x)) \Leftrightarrow w \in h(esum) \Leftrightarrow x \cdot w \in h(osum) \Leftrightarrow w \in \delta^{Pow}(h(osum))(x).$$

*Proof of (4).* Since  $\epsilon \in h(esum)$ ,  $\beta^{eo}(esum) = 1 = \beta^{Pow}(h(esum))$ . Since  $\epsilon \notin h(osum)$ ,  $\beta^{eo}(osum) = 0 = \beta^{Pow}(h(osum))$ .  $\square$

**Exercise 18** Let  $\mathcal{A}$  be the following  $DAut(\{a, b\}, 2)$ -algebra:

$$\begin{aligned}\mathcal{A}_{state} &= \{q, q_a, q_b, q_{ab}\}, \\ \beta^{\mathcal{A}} &= \lambda st. if\ st = qb\ then\ 1\ else\ 0,\end{aligned}$$

$$\begin{aligned}\delta^{\mathcal{A}}(q)(a) &= q_a, \quad \delta^{\mathcal{A}}(q)(b) = q_b, \quad \delta^{\mathcal{A}}(q_a)(a) = q, \quad \delta^{\mathcal{A}}(q_a)(b) = q_{ab}, \\ \delta^{\mathcal{A}}(q_b)(a) &= q_{ab}, \quad \delta^{\mathcal{A}}(q_b)(b) = q, \quad \delta^{\mathcal{A}}(q_{ab})(a) = q_b, \quad \delta^{\mathcal{A}}(q_{ab})(b) = q_a.\end{aligned}$$



Define  $h : \mathcal{A} \rightarrow \text{Pow}(\{a, b\})$  as follows

$$\begin{aligned} h(q) &= \{w \in \{a, b\}^* \mid \exists i, j \in \mathbb{N} : \#a(w) = 2*i \wedge \#b(w) = 2*j + 1\}, \\ h(q_a) &= \{w \in \{a, b\}^* \mid \exists i, j \in \mathbb{N} : \#a(w) = 2*i + 1 \wedge \#b(w) = 2*j + 1\}, \\ h(q_b) &= \{w \in \{a, b\}^* \mid \exists i, j \in \mathbb{N} : \#a(w) = 2*i \wedge \#b(w) = 2*j\}, \\ h(q_{ab}) &= \{w \in \{a, b\}^* \mid \exists i, j \in \mathbb{N} : \#a(w) = 2*i + 1 \wedge \#b(w) = 2*j\}. \end{aligned}$$

Prove that  $(\mathcal{A}, q)$  accepts  $h(q)$  by showing that  $h$  is  $D\text{Aut}(\{a, b\}, 2)$ -homomorphic—  
analogously to Example EOSUM.  $\square$

By Corollary DESMIN,

- the minimal realization of a behavior function  $f : X^* \rightarrow Y$  coincides with the least invariant  $\langle f \rangle$  of  $Beh(X, Y)$  that contains  $f$ ,
- the minimal acceptor of a word language  $L \subseteq X^*$  coincides with the least invariant  $\langle L \rangle$  of  $\text{Pow}(X)$  that contains  $L$ . Its final states are the languages of  $\langle L \rangle$  that contain  $(\cdot)$ .

It has been shown that for all  $t \in T_{Reg(X)}$ ,  $\langle t \rangle$  is finite ([36], Thm. 4.3 (a); [144], section 5; [79], Lemma 8). Hence, if combined with coinductive proofs of state equivalence, the stepwise construction of  $\langle t \rangle$  can be turned into a construction of a minimal acceptor of the **language of  $t$** —thus avoiding the traditional detour from a given automaton, its determinization (powerset construction) and subsequent minimization (see [153], section 4).

This fact allows us to build generic parsers for all regular languages upon  $\delta^{Bro(\langle \rangle)}$  and  $\beta^{Bro(\langle \rangle)}$  and to extend them to parsers for context-free languages by simply incorporating the respective grammar rules (see [125], chapter 17 and 18).

## $\Sigma$ -flowcharts

While terms denote both objects and computations, coterms denote only objects insofar as—intuitively—the behaviour a coterminus represents comprises all possible results of experiments (sequences of destructor applications) with a single element of a (final) model. So what is the counterpart of terms if these are regarded as computations, but its constructors are replaced by destructors?

“Destructor computations” come as various kinds of (data or control) *flow graphs* where the term “graph” indicates that cycles in the data flow are a frequent part of such computations. Here we choose the term “flowchart” because, indeed, destructor computations may be infinite, but their use for describing with iterative control structures is a topic of a later chapter (13).

We define flowcharts like terms, but with product and sum types and their respective ingredients exchanged:

Let  $\Sigma = (S, \mathcal{I}, F)$  be a destructive polynomial signature and  $V$  be an  $S$ -sorted set of “variables”.

$\Sigma$ -flowcharts are trees whose inner nodes are labelled with destructors or (indices of) projections, whose leaves are labelled with variables and whose edges are labelled with (indices of) injections. Hence they are dual to  $\Sigma$ -terms insofar as sums and products are exchanged here. More precisely:

The set  $\overline{CT}_\Sigma(V)$  of  **$\Sigma$ -flowcharts over  $V$**  is the greatest  $\mathcal{T}_p(S, \mathcal{I})$ -sorted set

$$M \subseteq \mathcal{I}^* \rightarrow (\mathcal{I} \cup F \cup V) + 1$$

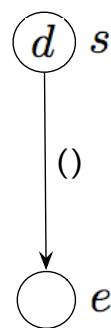
of labelled trees over  $(\mathcal{I}, \mathcal{I} \cup F \cup V)$  such that the following conditions hold true:

- For all  $s \in S$  and  $t \in M_s$ ,  $t \in V_s$  or there are  $d : s \rightarrow e \in F$  and  $u \in M_e$  such that  $t = d(u)$ , (1)
- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t \in M_e$  there are  $i \in I$  and  $u \in M_{e_i}$  such that  $t = i(u)$ . (2)
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t \in M_e$  there is  $(t_i)_{i \in I} \in \bigtimes_{i \in I} M_{e_i}$  such that  $t = ()\{i \rightarrow t_i \mid i \in I\}$ . (3)

$\overline{T}_\Sigma(V)$  denotes the least  $\mathcal{T}_p(S, \mathcal{I})$ -sorted set  $M$  of well-founded labelled trees over  $(\mathcal{I}, \mathcal{I} \cup F \cup V)$  such that the following conditions hold true:

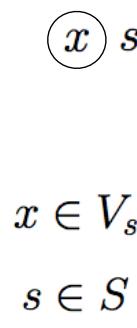
- for all  $d : s \rightarrow e \in F$  and  $u \in M_e$ ,  $d(u) \in M_s$ , (4)
- for all  $s \in S$ ,  $V_s \subseteq M_s$ , (5)
- for all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in M_{e_i}$ ,  $i(t) \in M_e$ , (6)
- for all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $(t_i)_{i \in I} \in \bigtimes_{i \in I} M_{e_i}$ ,  
 $()\{i \rightarrow t_i \mid i \in I\} \in M_e$ . (7)

Remember that every leaf of a *term* is labelled with either a variable (regarded as an *entrance* to the term) or an element of a primitive type. However, due to the above exclusion of primitive types from destructor targets, all leaves of a *flowchart* are labelled with variables and regarded as *exits* from the flowchart.

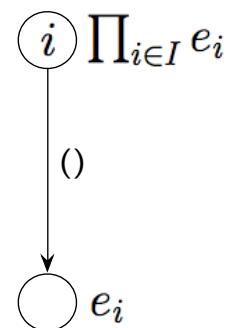


$$d : s \rightarrow e \in F$$

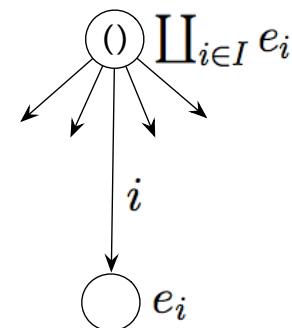
(1/4)



$$\begin{aligned} x &\in V_s \\ s &\in S \end{aligned}$$



(2/6)



(3/7)

In contrast to  $\Sigma$ -terms,  $\Sigma$ -flowcharts need not form an algebra. Hence we first define substitutions and instantiations of flowcharts and, secondly, their folding in algebras as different process, which—in contrast to term substitution—is not a special case of folding.

Given a further  $S$ -sorted set  $O$  of variables—called *output variables*, an  $S$ -sorted function  $g : V \rightarrow \overline{CT_\Sigma}(O)$  is called a **flowchart substitution** of  $V$ . The **flowchart instantiation**

$$g^* : \overline{T_\Sigma}(V) \rightarrow \overline{CT_\Sigma}(O)$$

by  $g$  is the  $\mathcal{T}_p(S, \mathcal{I})$ -sorted function that is defined inductively as follows:

- For all  $s \in S$  and  $x \in V_s$ ,  $g_s^*(x) = g_s(x)$ . (1)

- For all  $s \in \mathcal{I}$ ,  $g_s^*(s) = s$ . (2)

- For all  $d : s \rightarrow e \in F$  and  $t \in \overline{T_\Sigma}(V)_e$ ,  $g_s^*(d(t)) = d(g_e^*(t))$ . (3)

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in \overline{T_\Sigma}(V)_{e_i}$ ,

$$g_e^*(i(t)) = i(g_{e_i}^*(t)). \quad (4)$$

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $t = ()\{i \rightarrow t_i \mid i \in I\} \in \overline{T_\Sigma}(V)_e$  and  $i \in I$ ,

$$g_e^*(t) = ()\{i \rightarrow g_{e_i}^*(t_i) \mid i \in I\}. \quad (5)$$

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $O \times A =_{def} (O_s \times A_s)_{s \in S}$ . An  $S$ -sorted function

$$g = (g_s : V_s \rightarrow (O \times A)^{A_s})_{s \in S}$$

(where  $O \times A$  denotes the  $S$ -sorted product of  $O$  and  $A$ ) is called a **flowchart valuation of  $V$  in  $(A, O)$** . From now on,  $V \rightarrow (O \times A)^A$  denotes the set of flowchart valuations of  $V$  in  $(A, O)$ .

For instance,  $\text{return}_V =_{def} \lambda x. \lambda a. (x, a)$  is a flowchart valuation of  $V$  in  $(A, V)$ . It agrees with the unit of the reader/writer adjunction in  $\text{Set}^S$  (see section 19.8).

Given a flowchart valuation  $g : V \rightarrow (O \times A)^A$ , the **flowchart extension**

$$g^+ = (g_e : \overline{T_\Sigma}(V)_e \rightarrow (O \times A)^{A_e})_{e \in \mathcal{T}_p(S, \mathcal{I})}$$

of  $g$  to  $\overline{T_\Sigma}(V)$ , also called **flowchart folding**, is defined inductively as follows:

- For all  $s \in S$  and  $x \in V_s$ ,  $g_s^+(x) = g_s(x)$ .
- For all  $d : s \rightarrow e \in F$  and  $t \in \overline{T_\Sigma}(V)_e$ ,  $g_s^+(d(t)) = g_e^+(t) \circ d^A$ .
- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in \overline{T_\Sigma}(V)_{e_i}$ ,

$$g_e^+(i(t)) = g_{e_i}^+(t) \circ \pi_i.$$

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t = ()\{i \rightarrow t_i \mid i \in I\} \in \overline{T_\Sigma}(V)_e$ ,

$$g_e^+(t) = [g_{e_i}^+(t_i)]_{i \in I}.$$

Given  $s \in S$ ,  $t, t' \in \overline{T_\Sigma}(V)_s$ ,  $g : V \rightarrow (O \times A)^A$  solves the (flowchart) equation  $t = t'$  in  $\mathcal{A}$ , written as  $\mathcal{A} \models_g t = t'$ , if  $g^+(t) = g^+(t')$ .

$\mathcal{A}$  satisfies  $t = t'$ , written as  $\mathcal{A} \models t = t'$ , if all flowchart valuations of  $V$  in  $Val^A$  solve  $t = t'$  in  $\mathcal{A}$ .

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $e \in \mathcal{T}_p(S, \mathcal{I})$ .  $t \in \overline{T_\Sigma}(V)_e$  and  $\mathcal{A}$  define the function

$$t^A =_{def} return_V^+(t) : A_e \rightarrow V \times A.$$

## Lemma SUBSTF

Let  $\Sigma, V, O, \mathcal{A}, A$  be as above. For all flowchart substitutions  $g : V \rightarrow \overline{T_\Sigma}(O)$  and flowchart valuations  $h$  of  $O$  in  $(A, O)$ ,

$$(h^+ \circ g)^+ = h^+ \circ g^*.$$

*Proof.* Let  $t \in \overline{T_\Sigma}(V)$ . We show

$$(h^+ \circ g)^+(t) = h^+(g^*(t)) \tag{6}$$

by induction on  $t$ .

*Case 1.*  $t \in V$ . Then  $(h_\omega^+ \circ g)^+(t) = h_\omega^+(g(t)) = h_\omega^+(g^*(t))$ .

*Case 2.*  $t = d(u)$  for some  $d : s \rightarrow e \in D$  and  $u \in \overline{T_\Sigma}(V)$ . Then

$$\begin{aligned} (h^+ \circ g)^+(t) &= (h^+ \circ g)^+(u) \circ d^\mathcal{A} \stackrel{\text{ind.}}{=} h^+(g^*(u)) \circ d^\mathcal{A} = h^+(d(g^*(u))) \\ &= h^+(g^*(d(u))) = h^+(g^*(t)). \end{aligned}$$

*Case 3.*  $t = i(u)$  for some  $i \in I$ ,  $I \in \mathcal{I}$ ,  $u \in \overline{T_\Sigma}(V)_e$  and  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then (5) follows analogously to Case 2 with  $i$  instead of  $d$  and  $\pi_i$  instead of  $d^\mathcal{A}$ .

Case 4.  $t = ()\{i \rightarrow t_i \mid i \in I\}$  for some  $I \in \mathcal{I}$ ,  $(t_i)_{i \in I} \in \mathsf{X}_{i \in I} \overline{T_\Sigma}(V)_{e_i}$  and  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then

$$\begin{aligned} (h^+ \circ g)^+(t) &= (h^+ \circ g)^+((())\{i \rightarrow t_i \mid i \in I\}) = [(h^+ \circ g)^+(t_i)]_{i \in I} \stackrel{\text{ind. hyp.}}{=} [h^+(g^*(t_i))]_{i \in I} \\ &= [h^+(g^*(t_i))]_{i \in I} = h^+((())\{i \rightarrow g^*(t_i) \mid i \in I\}) = h^+(g^*(t)). \end{aligned} \quad \square$$

## From flowcharts to terms

Inspired by the recent literature on *comodels* and their use for *effectful programming* (see, e.g., [134, 129, 27]), we call a destructive signature  $\Sigma = (S, \mathcal{I}, F)$  a **comodel signature** if  $\Sigma$  is finitary and for all  $d : s \rightarrow s_1 + \dots + s_n \in F$ ,  $s_1, \dots, s_n \in S$ .

Let  $\Sigma = (S, \mathcal{I}, F)$  be a comodel signature.

For all  $s \in S$ ,  $\bar{s} =_{\text{def}} s$ . For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $\bar{e} =_{\text{def}} \prod_{i \in I} e_i$ .  $\bar{F} =_{\text{def}} \{\bar{d} : \bar{e} \rightarrow s \mid d : s \rightarrow e \in F\}$  and  $\bar{\Sigma} =_{\text{def}} (S, \bar{F})$ . Hence  $\bar{\Sigma}$  is constructive.

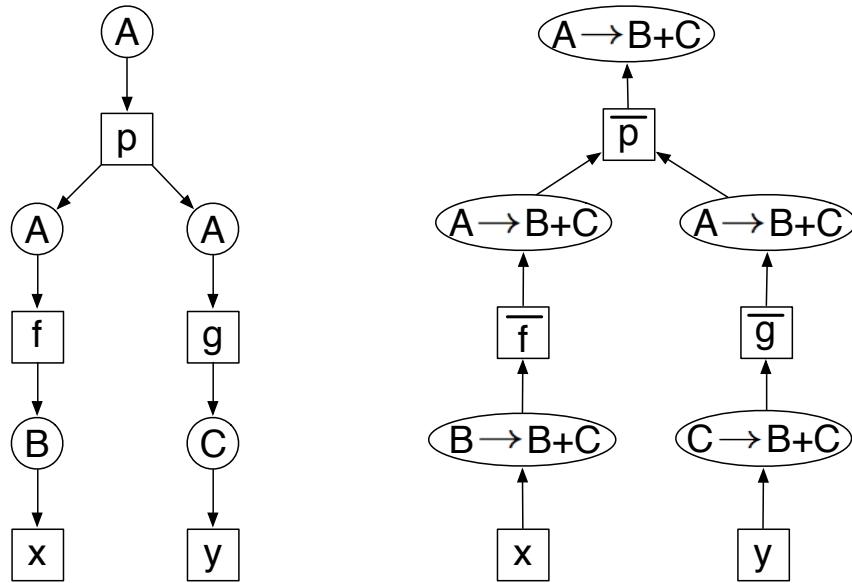
A  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  induces the  $\bar{\Sigma}$ -algebra  $\mathcal{A}_V$  with carrier  $A_V$  that is defined as follows:

- For all  $s \in S$ ,  $A_{V,s} = A_s \rightarrow V \times A$ .
- For all  $d : s \rightarrow s' \in F$ ,  $\bar{d}^{\mathcal{A}_V} = \text{lift}_{V \times A}(d^{\mathcal{A}}) : A_{V,s'} \rightarrow A_{V,s}$  (see [Sums](#)).
- For all  $d : s \rightarrow \coprod_{i \in I} s_i \in F$ ,  $\bar{d}^{\mathcal{A}_V} = \text{lift}_{V \times A}(d^{\mathcal{A}}) : \prod_{i \in I} A_{V,s_i} \rightarrow A_{V,s}$ .

Given a  $\Sigma$ -flowchart  $t$ , the  $\bar{\Sigma}$ -term  $\bar{t}$  is obtained from  $t$  by replacing every node label  $d \in F$  with  $\bar{d}$ .

## Example

Somewhat simplified  $\Sigma$ -flowchart and its corresponding  $\bar{\Sigma}$ -term, extended by the source or target types (in ovals) of the involved functions:



Let  $\varphi : A \rightarrow 2$  and  $p : A \rightarrow A + A$  be defined as follows:  $p(a) = \iota_1(a) \Leftrightarrow \varphi(a) = 1$ . Moreover, let  $f : A \rightarrow B$ ,  $g : A \rightarrow C$  and  $V = \{x, y\}$ . Then for all  $h_1, h_2 : A \rightarrow B + C$ ,  $h_3 : B \rightarrow B + C$ ,  $h_4 : C \rightarrow B + C$  and  $a \in A$ ,

$$\begin{aligned}
 \bar{p}(h_1, h_2)(a) &= ([h_1, h_2] \circ p)(a) = [h_1, h_2](p(a)) = \begin{cases} h_1(a) & \text{if } p(a) = \iota_1(a) \\ h_2(a) & \text{if } p(a) = \iota_2(a) \end{cases} \\
 &= \begin{cases} h_1(a) & \text{if } \varphi(a) = 1 \\ h_2(a) & \text{if } \varphi(a) = 0 \end{cases}, \quad \bar{f}(h_3)(a) = (h_3 \circ f)(a) = h_3(f(a)), \\
 \bar{g}(h_4)(a) &= (h_4 \circ g)(a) = h_4(g(a)).
 \end{aligned}$$

□

More examples are given in the section on [flowchart equations](#).

Obviously, every flowchart valuation  $g$  of  $V$  in  $A$  is a term valuation of  $V$  in  $A_V$  and vice versa (see [Term folding](#)). However,  $g^+$  evaluates flowcharts, while  $g^*$  evaluates terms.

The observation allows us to adapt [129], Lemma 6.2 to comodel signatures as defined above:

## Lemma FLOW

Let  $\Sigma$  be a comodel signature,  $\mathcal{A}, A$  be as above and  $g$  be a flowchart valuation of  $V$  in  $(A, O)$ .

- (1) For all  $t \in \overline{T_\Sigma}(V)$ ,  $g^+(t) = g^*(\bar{t})$ .
- (2) For all  $t, t' \in \overline{T_\Sigma}(V)$ ,  $g$  solves  $t = t'$  in  $\mathcal{A}$  iff  $g$  solves  $\bar{t} = \bar{t}'$  in  $\mathcal{A}_{V'}$ .

*Proof of (1) by induction on  $t$ .*

*Case 1:*  $t \in V$ . Then  $g^+(t) = g(t) = g^*(t) = g^*(\bar{t})$ .

*Case 2:*  $t = d(u)$  for some  $d : s \rightarrow s' \in F$  with  $s' \in S$  and  $u \in \overline{T_\Sigma}(V)_{s'}$ . Then

$$g^+(t) = g^+(d(u)) = g^+(u) \circ d^{\mathcal{A}} \stackrel{\text{ind. hyp.}}{=} g^*(\bar{u}) \circ d^{\mathcal{A}} = \bar{d}^{\mathcal{A}_{V'}}(g^*(\bar{u})) = g^*(\bar{d}(\bar{u})) = g^*(\bar{t}).$$

Case 3:  $t = d(\emptyset \{i \rightarrow t_i \mid i \in I\})$  for some  $d : s \rightarrow \coprod_{i \in I} s_i \in F$  and  $(t_i)_{i \in I} \in \mathsf{X}_{i \in I} \overline{T_\Sigma}(V)_{s_i}$ . Then

$$\begin{aligned} g^+(t) &= g^+(d(\emptyset \{i \rightarrow t_i \mid i \in I\})) = g^+(\emptyset \{i \rightarrow t_i \mid i \in I\}) \circ d^\mathcal{A} = [g^+(t_i)]_{i \in I} \circ d^\mathcal{A} \\ &\stackrel{\text{ind. hyp.}}{=} [g^*(\bar{t}_i)]_{i \in I} \circ d^\mathcal{A} = [\pi_i(g^*(\emptyset \{i \rightarrow \bar{t}_i \mid i \in I\}))]_{i \in I} \circ d^\mathcal{A} \\ &= \bar{d}^{\mathcal{A}_{V'}}(g^*(\emptyset \{i \rightarrow \bar{t}_i \mid i \in I\})) = g^*(\bar{d}(\emptyset \{i \rightarrow \bar{t}_i \mid i \in I\})) = g^*(\bar{t}). \end{aligned}$$

(2) follows immediately from (1).  $\square$

## $\Sigma$ -formulas: Syntax

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature.  $\Sigma$  is called **Kripke** if  $S$  includes the sorts

*state, label, atom*

and  $F$  includes the arrows

$$\begin{aligned} \textit{inits} &: 1 \rightarrow \mathcal{P}_\omega(\textit{state}), \\ \textit{trans} &: \textit{state} \rightarrow \mathcal{P}_\omega(\textit{state}), \\ \textit{transL} &: \textit{state} \times \textit{label} \rightarrow \mathcal{P}_\omega(\textit{state}), \\ \textit{value} &: \textit{atom} \rightarrow \mathcal{P}_\omega(\textit{state}), \\ \textit{valueL} &: \textit{atom} \times \textit{label} \rightarrow \mathcal{P}_\omega(\textit{state}). \end{aligned}$$

Kripke signatures allow us to specify almost any kind of Kripke-like structure and incorporate not only the whole range of higher-order functions,  $\lambda$ -expressions, etc., but also logic operators known from CTL (computation tree logic), the modal  $\mu$ -calculus, from query languages like SQL, relational algebra, description logic or XPath (see, e.g., [57, 152, 99]) and also from dynamic logic that admits the verification of imperative programs with iteration and has been reformulated coalgebraically in [105], chapter 7.

The integration of these approaches reveals many semantical overlappings. For example, many operators used in one approach are simply compositions of operators used in other approaches. E.g., basic CTL operators are special cases of the existential or universal quantification used in description logic, while the “advanced” CTL operators can be reduced to fixpoint operators known from the modal  $\mu$ -calculus.

While every formula of CTL or the modal  $\mu$ -calculus denotes a *state predicate*, i.e., a set of states, description logic, XPath and other languages for querying tree-like documents provide formulas representing binary relations between states. The tables of a relational database can also be regarded as states. Here each state unfolds to a set of further states, which are names for the rows of the table. A row itself is modelled as a partial function that maps labels representing *attributes* to other states representing attribute values.

Since  $\Sigma$ -formulas involve functional expressions, states, labels and atoms may be highly structured so that transition and valuation functions (see above) may be specified by subtle case distinctions based on pattern matching. Moreover, further operators on sets of or relations between states as well as on relational databases could be added. Even *temporal* logics that deal with streams or infinite trees of states could be integrated, perhaps by equipping  $\Sigma$  with suitable (polynomial or quantitative) destructors.

In contrast to  $\Sigma$ -arrows (see chapter 8),  $\Sigma$ -formulas may contain variables of any type over  $(S, \mathcal{I})$ . Whereas fixpoint operators occurring in  $\Sigma$ -formulas always bind a single variable of a powerset type,  $\lambda$ -abstraction occurring in  $\Sigma$ -formulas may bind several variables of any type of  $\mathcal{T}(S, \mathcal{I})$ . Of course, classical (many-sorted) first-order logic with single-variable quantification is also included.

$\Sigma$ -terms and  $\Sigma$ -flowcharts also contain variables.  $\Sigma$ -terms have *polynomial* types and are composed of constructors and thus always denote (parametrized) *objects* of initial algebras (see chapter 9). Hence a  $\Sigma$ -term is *both* a particular  $\Sigma$ -formula  $\varphi$  and the result of interpreting  $\varphi$  in an initial term algebra.  $\Sigma$ -flowcharts represent *functions* into sum types, which may also be specified as  $\Sigma$ -arrows or (variable-free)  $\Sigma$ -formulas. Variables of flowcharts denote exits for output, not place-holders for input as  $\Sigma$ -terms and  $\Sigma$ -formulas do.

Finally,  $\Sigma$ -coterms denote *behaviours*, which are—like  $\Sigma$ -terms—particular *object* representations. But—in contrast to  $\Sigma$ -terms—they have no *functional* interpretation.

Let  $\Sigma = (S, \mathcal{I}, F)$  be a Kripke signature and  $V$  be a  $\mathcal{T}(S, \mathcal{I})$ -sorted set of variables. The  $\mathcal{T}(S, \mathcal{I})$ -sorted  $Fo_\Sigma(V)$  of  **$\Sigma$ -formulas over  $V$**  is inductively defined as follows:

Let  $e, e', e'', e_1, \dots, e_n \in \mathcal{T}(S, \mathcal{I})$ ,  $I \subseteq \mathcal{I}$ ,  $C$  be the set of constructors of  $F$ ,  $C\Sigma = (S, \mathcal{I}, C)$  and  $\text{row} = (\text{state} + 1)^{\text{label}}$ .

- $\mathcal{I} \cup V \cup Arr_\Sigma \cup T_{C\Sigma}(V) \subseteq Fo_\Sigma(V)$ . (indices, variables,  $\Sigma$ -arrows and  $C\Sigma$ -terms)
- $Fo_\Sigma(V)_e \subseteq Fo_\Sigma(V)_{1 \rightarrow e}$ .
- $\text{ite} : 2 \times e \times e \rightarrow e \in Fo_\Sigma(V)$ . (if-then-else)
- For all  $(f_i : e_i \rightarrow e)_{i \in I} \in Fo_\Sigma(V)^I$ ,  $[f_i]_{i \in I} : \coprod_{i \in I} e_i \rightarrow e \in Fo_\Sigma(V)$ . (sum extensions)
- For all  $(f_i : e \rightarrow e_i)_{i \in I} \in Fo_\Sigma(V)^I$ ,  
 $\langle f_i \rangle_{i \in I} : e \rightarrow \prod_{i \in I} e_i \in Fo_\Sigma(V)$ . (product extensions)
- For all  $(t_i : e_i)_{i \in I} \in Fo_\Sigma(V)^I$ ,  $(t_i)_{i \in I} : \prod_{i \in I} e_i \in Fo_\Sigma(V)$ . (tupling)
- $\text{single} : e \rightarrow \mathcal{P}_\omega(e)$ .
- For all  $f : e \rightarrow e' \in Fo_\Sigma(V)$ ,  $\text{map}(f) : \mathcal{P}(e) \rightarrow \mathcal{P}(e') \in Fo_\Sigma(V)$ . (mapping)

- $joinM : \mathcal{P}(\mathcal{P}(e)) \rightarrow \mathcal{P}(e) \in Fo_{\Sigma}(V)$ . (monadic multiplication)
- For all  $p : e \rightarrow 2 \in Fo_{\Sigma}(V)$ ,  $filter(p) : \mathcal{P}(e) \rightarrow \mathcal{P}(e) \in Fo_{\Sigma}(V)$ . (filtering)
- For all  $f : e \times e' \rightarrow e \in Fo_{\Sigma}(V)$ ,  $foldl(f) : e \rightarrow (e'^* \rightarrow e) \in Fo_{\Sigma}(V)$ . (list folding)
- For all  $sucs : e \rightarrow \mathcal{P}(e)$ ,  $f : e \times e' \rightarrow \mathcal{P}(e) \in Fo_{\Sigma}(V)$ , (monadic list folding)

$$foldlMS(sucs)(f) : e \rightarrow (e'^* \rightarrow \mathcal{P}(e)) \in Fo_{\Sigma}(V).$$

- For all  $(e_i)_{i \in I} \in \mathcal{T}(S, \mathcal{I})^I$ ,  $x \in \mathsf{X}_{i \in I} V_{e_i}$  and  $t : e \in Fo_{\Sigma}(V)$ ,  $\lambda x.t : \prod_{i \in I} e_i \rightarrow e \in Fo_{\Sigma}(V)$ . ( $\lambda$ -abstraction)
- For all  $f : e \rightarrow e'$ ,  $t : e \in Fo_{\Sigma}(V)$ ,  $f(t) : e' \in Fo_{\Sigma}(V)$ . ( $\lambda$ -application)
- $=, \neq : e \times e \rightarrow 2 \in Fo_{\Sigma}(V)$
- For all  $e \in \{2, \mathcal{P}(e')\}$ ,  $true, false : e, \neg : e \rightarrow e \in Fo_{\Sigma}(V)$ .
- For all  $e \in \{2, \mathcal{P}(e')\}$ ,  $x \in V$  and  $\varphi : e \in Fo_{\Sigma}(V)$ ,  $\forall x \varphi, \exists x \varphi : e \in Fo_{\Sigma}(V)$ .
- $rel2fun : \mathcal{P}(e \times e') \rightarrow (e \rightarrow \mathcal{P}(e')) \in Fo_{\Sigma}(V)$ .
- For all  $p : e \rightarrow 2 \in Fo_{\Sigma}(V)$ ,  $sat(p) : \mathcal{P}(e) \in Fo_{\Sigma}(V)$ .
- For all  $r : e \times e' \rightarrow 2 \in Fo_{\Sigma}(V)$ ,  $sat(r) : \mathcal{P}(e \times e') \in Fo_{\Sigma}(V)$ .
- For all  $p : \mathcal{P}(e') \rightarrow 2 \in Fo_{\Sigma}(V)$ ,  $select(p) : \mathcal{P}(e \times e') \rightarrow \mathcal{P}(e) \in Fo_{\Sigma}(V)$ .
- $inv : \mathcal{P}(e \times e') \rightarrow \mathcal{P}(e' \times e) \in Fo_{\Sigma}(V)$ .
- $\pi_1 : \mathcal{P}(e \times e') \rightarrow \mathcal{P}(e)$ ,  $\pi_2 : \mathcal{P}(e \times e') \rightarrow \mathcal{P}(e') \in Fo_{\Sigma}(V)$ .

- $* : \mathcal{P}(e) \times \mathcal{P}(e) \rightarrow \mathcal{P}(e^2) \in \text{Fo}_\Sigma(V)$ .
- $/ : \mathcal{P}(e \times e') \times \mathcal{P}(e') \rightarrow \mathcal{P}(e) \in \text{Fo}_\Sigma(V)$ .
- $; : \mathcal{P}(e \times e') \times \mathcal{P}(e' \times e'') \rightarrow \mathcal{P}(e \times e'') \in \text{Fo}_\Sigma(V)$ .
- $\exists, \forall : \mathcal{P}(e \times e') \rightarrow (\mathcal{P}(e') \rightarrow \mathcal{P}(e)) \in \text{Fo}_\Sigma(V)$ .
- For all  $x : e \in V$  and  $\varphi : e \in \text{Fo}_\Sigma(V)$ ,  
 $\mu x.\varphi, \nu x.\varphi : e \in \text{Fo}_\Sigma(V)$ . (least and greatest predicates)
- For all  $x : e \times e' \in V$  and  $\rho : e \times e' \in \text{Fo}_\Sigma(V)$ ,  
 $\mu x.\rho, \nu x.\rho : e \times e' \in \text{Fo}_\Sigma(V)$ . (least and greatest relations)
- $\sim : \mathcal{P}(\text{state}^2) \in \text{Fo}_\Sigma(V)$ . (behavioural equivalence)
- $\text{labels} : \mathcal{P}_\omega(\text{label}) \in \text{Fo}_\Sigma(V)$ .
- $\text{preds} : \text{state} \rightarrow \mathcal{P}_\omega(\text{state}) \in \text{Fo}_\Sigma(V)$ ,  
 $\text{predsL} : \text{state} \rightarrow (\text{label} \rightarrow \mathcal{P}_\omega(\text{state})) \in \text{Fo}_\Sigma(V)$ . (predecessors)
- $\text{out} : \text{state} \rightarrow \mathcal{P}_\omega(\text{atom}) \in \text{Fo}_\Sigma(V)$ ,  
 $\text{outL} : \text{state} \rightarrow (\text{label} \rightarrow \mathcal{P}_\omega(\text{atom})) \in \text{Fo}_\Sigma(V)$ . (output)
- $\text{child} : \mathcal{P}_\omega(\text{label}) \rightarrow \mathcal{P}_\omega(\text{state}^2) \in \text{Fo}_\Sigma(V)$ .
- $\text{mkrow} : \text{state} \rightarrow \text{row} \in \text{Fo}_\Sigma(V)$ .
- $\text{projectrow} : \mathcal{P}_\omega(\text{label}) \rightarrow (\text{row} \rightarrow \text{row}) \in \text{Fo}_\Sigma(V)$ .
- For all  $p : 2 \in \text{Fo}_\Sigma(V)$ ,  $\text{mkinstance}(p) : \text{row} \rightarrow 2 \in \text{Fo}_\Sigma(V)$ .

- $\wedge, \vee, -, \times, / : \mathcal{P}_\omega(\text{row}) \times \mathcal{P}_\omega(\text{row}) \rightarrow \mathcal{P}_\omega(\text{row}) \in \text{Fo}_\Sigma(V)$ .
- $njoin : \mathcal{P}_\omega(\text{row}) \rightarrow (\mathcal{P}_\omega(\text{row}) \rightarrow \mathcal{P}_\omega(\text{row})) \in \text{Fo}_\Sigma(V)$ .

## Derived $\Sigma$ -formulas

Let  $e, e', e'', e_1, \dots, e_n \in \mathcal{T}(S, \mathcal{I})$ ,  $I \subseteq \mathcal{I}$  and  $\text{row} = (\text{state} + 1)^{\text{label}}$ .

- For all  $f : e \rightarrow e', g : e' \rightarrow e'' \in \text{Fo}_\Sigma(V)$  and some  $x : e \in V$ , (composition)

$$g \circ f = \lambda x. g(f(x)) : e \rightarrow e''.$$

- For all  $f : e \rightarrow \mathcal{P}(e') \in \text{Fo}_\Sigma(V)$ , (monadic extension)

$$\text{joinMap}(f) = \text{joinM} \circ \text{map}(f) : \mathcal{P}(e) \rightarrow \mathcal{P}(e').$$

- For all  $f : e \rightarrow \mathcal{P}(e')$ ,  $g : e' \rightarrow \mathcal{P}(e'') \in \text{Fo}_\Sigma(V)$ , (monadic composition)

$$g \lll f = \text{joinMap}(g) \circ f : e \rightarrow \mathcal{P}(e'').$$

- For all  $f : e \times e' \rightarrow e''$ ,  $g : e \rightarrow (e' \rightarrow e'') \in \text{Fo}_\Sigma(V)$  and some  $x : e$ ,  $y : e' \in V$ ,

$$\text{curry}(f) = \lambda x. \lambda y. f(x, y) : e \rightarrow (e' \rightarrow e''),$$

$$\text{uncurry}(g) = \lambda(x, y). g(x)(y) : e \times e' \rightarrow e''.$$

- For all  $f : e \rightarrow (e' \rightarrow e'') \in \text{Fo}_\Sigma(V)$  and some  $x : e, y : e' \in V$ ,

$$\text{flip}(f) = \lambda y. \lambda x. f(x)(y) : e' \rightarrow (e \rightarrow e'').$$

- $\text{actL} = \text{flip}(\text{curry}(\text{transL})) : \text{label} \rightarrow (\text{state} \rightarrow \mathcal{P}_\omega(\text{state}))$ . (labels as actions)

- For all  $(f_i : e_i \rightarrow e'_i)_{i \in I} \in \text{Fo}_\Sigma(V)^I$ , (sums and products)

$$\coprod_{i \in I} f_i = [\iota_i \circ f_i]_{i \in I} : \coprod_{i \in I} e_i \rightarrow \coprod_{i \in I} e'_i \quad \text{and} \quad \prod_{i \in I} f_i = \langle f_i \circ \pi_i \rangle_{i \in I} : \prod_{i \in I} e_i \rightarrow \prod_{i \in I} e'_i.$$

- For all  $e \in \mathcal{T}_p(S, \mathcal{I})$  and  $T = \{f_s : e_s \rightarrow e'_s \mid s \in S\} \subseteq \text{Fo}_\Sigma(V)$ , (type instances)

$$T_e : e[e_s/s \mid s \in S] \rightarrow e[e'_s/s \mid s \in S]$$

is inductively defined as follows:

For all  $s \in S, i \in \mathcal{I}$  and  $(e_i)_{i \in I} \in \mathcal{T}_p(S, \mathcal{I})^I, e \in \mathcal{T}_p(S, \mathcal{I})$ , commutative monoids  $(M, +, 0)$  and  $C \subseteq M$ ,

$$\begin{aligned} T_s &= f_s, \quad T_i = id_{\{i\}}, \\ T_{\prod_{i \in I} e_i} &= \prod_{i \in I} T_{e_i}, \quad T_{\coprod_{i \in I} e_i} = \coprod_{i \in I} T_{e_i}, \\ T_{(M \times e)_C^*} &= (id_M \times T_e)_C^*. \end{aligned}$$

- For all  $e \in \{2, \mathcal{P}(e')\}$  and  $\varphi, \psi : e \in \text{Fo}_\Sigma(V)$ ,

$$\varphi \Rightarrow \psi = \neg \varphi \vee \psi : e \quad \text{and} \quad \varphi - \psi = \varphi \wedge \neg \psi : e.$$

- For all  $e \in \{2, \mathcal{P}(e')\}$ ,

$$\text{and} = \text{foldl}(\wedge)(\text{true}) : e^* \rightarrow e \quad \text{and} \quad \text{or} = \text{foldl}(\vee)(\text{false}) : e^* \rightarrow e.$$

- For all  $f : e \rightarrow \mathcal{P}(e)$ ,  $\text{foldlM} = \text{foldlMS}(\text{single})$ .

- For all  $\varphi : \mathcal{P}(\text{state})$ ,

$$\text{successors}(\emptyset)(\varphi) = \mu x. (\varphi \vee \text{joinMap}(\text{trans})(x)) : \mathcal{P}(\text{state}).$$

- $\text{unfoldNDS} = \lambda st. (\text{out} <= < \text{foldlMS}(\text{successors}(\emptyset))(\text{transL})(st))$   
 $: \text{state} \rightarrow (\text{label}^* \rightarrow \mathcal{P}_\omega(\text{atom}))$ .

This reduces to

$$\lambda st. (\text{out} <= < \text{foldlM}(\text{transL})(st))$$

if the actual Kripke model  $\mathcal{A}$  (see below) satisfies  $\text{trans}^\mathcal{A} = \lambda st. \emptyset$  and to

$$\lambda st. (\text{out} \circ \text{foldl}(\text{transL})(st))$$

if  $\mathcal{A}$  is *deterministic*, i.e., for all  $a \in \mathcal{A}(\text{state})$ ,  $|\text{trans}^\mathcal{A}(a)| = 1$ .

- For all  $ts : \mathcal{P}_\omega(\text{label}) \in \text{Fo}_\Sigma(V)$ ,  $\text{sibling}(ts) = \text{child}(ts); \text{parent}(ts) : \mathcal{P}_\omega(\text{state}^2)$ .

- For all  $\varphi : \mathcal{P}(e)$ ,  $\psi : \mathcal{P}(e')$ ,  $\rho : \mathcal{P}(e \times e') \in \text{Fo}_\Sigma(V)$ ,

$$\varphi \ll \rho = \varphi \wedge \pi_1 \rho : \mathcal{P}(e) \quad \text{and} \quad \rho \gg \psi = \pi_2 \rho \wedge \psi : \mathcal{P}(e').$$

- $\text{parent} = \text{inv} \circ \text{child} : \mathcal{P}_\omega(\text{label}) \rightarrow \mathcal{P}_\omega(\text{state}^2)$ .

- For all  $ts : \mathcal{P}_\omega(\text{label}) \in \text{Fo}_\Sigma(V)$ ,

$$\begin{aligned} \langle ts \rangle &= \exists(\text{child}(ts)) : \mathcal{P}(\text{state}) \rightarrow \mathcal{P}(\text{state}), \\ [ts] &= \forall(\text{child}(ts)) : \mathcal{P}(\text{state}) \rightarrow \mathcal{P}(\text{state}). \end{aligned}$$

- For all  $\rho : \mathcal{P}(e^2) \in \text{Fo}_\Sigma(V)$  and  $x : \mathcal{P}(e^2) \in V$ , (transitive closure)

$$tcl(\rho) = \mu x.(\rho \vee (x; \rho)) : \mathcal{P}(e^2).$$

- $\text{descendant} = tcl \circ \text{child} : \mathcal{P}_\omega(\text{label}) \rightarrow \mathcal{P}(\text{state}^2)$ .

- $\text{ancestor} = tcl \circ \text{parent} : \mathcal{P}_\omega(\text{label}) \rightarrow \mathcal{P}(\text{state}^2)$ .

- For all  $ts : \mathcal{P}_\omega(\text{label}) \in \text{Fo}_\Sigma(V)$ ,

$$\text{related}(ts) = \text{ancestor}(ts); \text{ sibling}(ts); \text{ descendant}(ts) : \mathcal{P}(\text{state}^2).$$

- For all  $ts : \mathcal{P}_\omega(\text{label}) \in \text{Fo}_\Sigma(V)$  and  $\varphi : \mathcal{P}(\text{state})$ ,

$$\text{successors}(ts)(\varphi) = \mu x.(\varphi \vee \exists(\text{parent}(ts))(x)) : \mathcal{P}(\text{state}),$$

$$\text{allsuccessors}(\varphi) = \text{successors}(\emptyset)(\varphi) \vee \text{successors}(\text{labels})(\varphi) : \mathcal{P}(\text{state}).$$

- For all  $\varphi, \psi : \mathcal{P}(\text{state}) \in \text{Fo}_\Sigma(V)$  and some  $x : \mathcal{P}(\text{state}) \in V$ ,

$$\begin{aligned}
 EF(\varphi) &= \mu x. (\varphi \vee \langle \emptyset \rangle x) && (\varphi \text{ finally on some path}) \\
 &= \exists (\text{descendant}(\emptyset))(\varphi) : \mathcal{P}(\text{state}), \\
 AG(\varphi) &= \mu x. (\varphi \wedge [\emptyset]x) && (\varphi \text{ generally on all paths}) \\
 &= \forall (\text{descendant}(\emptyset))(\varphi) : \mathcal{P}(\text{state}), \\
 EG\varphi &= \nu x. (\varphi \wedge (\langle \emptyset \rangle x \vee [\emptyset]\text{false})) : \mathcal{P}(\text{state}), && (\varphi \text{ generally on some path}) \\
 AF\varphi &= \mu x. (\varphi \vee ([\emptyset]x \wedge \langle \emptyset \rangle \text{true})) : \mathcal{P}(\text{state}), && (\varphi \text{ finally on all paths}) \\
 \varphi EU\psi &= \mu x. (\psi \vee (\varphi \wedge \langle \emptyset \rangle x)) : \mathcal{P}(\text{state}), && (\varphi \text{ until } \psi \text{ on some path}) \\
 \varphi AU\psi &= \mu x. (\psi \vee (\varphi \wedge [\emptyset]x)) : \mathcal{P}(\text{state}). && (\varphi \text{ until } \psi \text{ on all paths})
 \end{aligned}$$

- $mktab = \text{map}(\text{mkrow}) \circ \text{trans} : \text{state} \rightarrow \mathcal{P}_\omega(\text{row})$ .
- $\text{project} = \text{map} \circ \text{projectrow} : \mathcal{P}_\omega(\text{row}) \rightarrow \mathcal{P}_\omega(\text{row})$ .
- For all  $p : 2 \in \text{Fo}_\Sigma(V)$ ,

$$\text{select}(p) = \text{filter}(\text{mkinstance}(p)) : \mathcal{P}_\omega(\text{row}) \rightarrow \mathcal{P}_\omega(\text{row}).$$

- For all  $p : \text{row} \rightarrow 2$ ,  $\vartheta : \mathcal{P}_\omega(\text{row}) \in \text{Fo}_\Sigma(V)$ ,

$$tjoin(p)(\vartheta) = \text{select}(p) \circ njoin(\vartheta) : \mathcal{P}_\omega(\text{row}).$$

Every  $\Sigma$ -formula is an expression where for all  $x \in V$  and  $z \in V_{\mathcal{P}(\text{state})} \cup V_{\mathcal{P}(\text{state}^2)}$ ,  $\lambda x, \exists x, \forall x, \mu z, \nu z$  are regarded as unary operators and every formula is a labelled tree, namely a function from  $(\mathcal{I} \cup \mathbb{N}_+)^*$  to  $(\mathcal{I} \cup F \cup Op \cup App) + 1$  where  $Op$  denotes the set of (basic or derived) operators introduced above and

$$App = \{\$\colon ((e \rightarrow e') \times e) \rightarrow e' \mid e, e' \in \mathcal{T}(S, \mathcal{I})\}$$

denotes the set of binary application operators that must be inserted into the expressions for regarding them as trees.

Let  $\varphi, \varphi_1, \dots, \varphi_n$  be  $\Sigma$ -formulas. In terms of their tree representation,  $w \in def(\varphi)$  has **positive** or **negative polarity** if the number of prefixes  $v$  of  $w$  with  $\varphi(v) = \neg$  is even or odd, respectively.

$w \in (\mathcal{I} \cup \mathbb{N}_+)^*$  is called an **occurrence of  $x \in \mathcal{I} \cup F \cup Op \cup App$  in  $\varphi$**  if  $\varphi(w) = x$ .

$occ(x, \varphi)$  denotes the set of occurrences of  $x$  in  $\varphi$ .

$x$  **occurs in  $\varphi$**  if  $occ(x, \varphi)$  is not empty.

A  $\varphi$ -occurrence  $w$  of  $x$  is **bound in  $\varphi$**  if  $\varphi(v) \in \{\lambda x, \forall x, \exists x, \mu x, \nu x\}$  for some prefix  $v$  of  $w$ .

$\text{bound}(x, \varphi)$  denotes the set of bound occurrences of  $x$  in  $\varphi$ .

$x \in V$  is a **free variable** of  $\varphi$  if  $\text{occ}(x, \varphi) \neq \text{bound}(x, \varphi)$ .

$\text{free}(\varphi)$  denotes the set of **free variables** of  $\varphi$ .

$\varphi$  is **closed** if  $\varphi$  does not contain free variables.

$\text{var}(\varphi)$  denotes the set of  $x \in V$  such that  $\text{occ}(x, \varphi) \neq \emptyset$ .

$\text{var}(\varphi_1, \dots, \varphi_n) =_{\text{def}} \text{var}(\varphi_1) \cup \dots \cup \text{var}(\varphi_n)$ .

## $\Sigma$ -formulas: Semantics

Let  $\Sigma$  be a Kripke signature,  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $V$  be a  $\mathcal{T}(S, \mathcal{I})$ -sorted set of variables such that  $A_{state}$  and  $A_{label}$  are finite.

The interpretation of quantitative types in  $\mathcal{A}$  is extended to exponential and powerset types are interpreted as expected: For all  $e, e' \in \mathcal{T}(S, \mathcal{I})$ ,  $A_{e \rightarrow e'} = (A_e \rightarrow A_{e'})$  and  $A_{\mathcal{P}(e)} = \mathcal{P}(A_e)$ .

$(init^{\mathcal{A}}, trans^{\mathcal{A}}, transL^{\mathcal{A}}, value^{\mathcal{A}}, valueL^{\mathcal{A}})$  is called the **Kripke model of  $\mathcal{A}$** .

A **formula valuation** of  $V$  in  $A$  is a function tuple  $g = (g_e : V_e \rightarrow A_e)_{e \in \mathcal{T}(S, \mathcal{I})}$  such that for all  $e \in \mathcal{T}(S, \mathcal{I})$  and  $x \in V_e$ ,  $g(x) \in A_e$ .

From now on,  $A^V$  denotes the set of formula valuations of  $V$  in  $A$ .

Given  $\varphi : e \in Fo_{\Sigma}(V)$ , the interpretation of  $\varphi$  in  $\mathcal{A}$  is given by the function  $\varphi^{\mathcal{A}} : A^V \rightarrow A_e$  that is inductively defined as follows:

Let  $g \in A^V$ ,  $Q = A_{state}$ ,  $L = A_{label}$ ,  $At = A_{atom}$ ,  $e, e', e'', e_1, \dots, e_n \in \mathcal{T}(S, \mathcal{I})$  and  $I \subseteq \mathcal{I}$ . If  $\varphi$  is closed, we write  $\varphi^{\mathcal{A}}$  instead of  $\varphi^{\mathcal{A}}(g)$  because  $\varphi^{\mathcal{A}}$  does not depend on  $g$ . For the definition of the powerset functor  $\mathcal{P}$ , see Preliminaries.

- For all  $i \in \mathcal{I}$ ,  $i^{\mathcal{A}} = i$
- For all  $x \in V$ ,  $x^{\mathcal{A}}(g) = g(x)$ .

- For all  $t \in Arr_\Sigma \cup T_{C\Sigma}(V)$ ,  $t^{\mathcal{A}}(g)$  is defined in chapter 9.
- For all  $a, b \in A_e$ ,  $ite^{\mathcal{A}}(1, a, b) = a$  and  $ite^{\mathcal{A}}(0, a, b) = b$ .
- For all  $(t_i : e_i \rightarrow e)_{i \in I} \in Fo_\Sigma(V)^I$ ,  $[t_i]_{i \in I}^{\mathcal{A}}(g) = [t_i^{\mathcal{A}}(g)]_{i \in I}$ .
- For all  $(t_i : e \rightarrow e_i)_{i \in I} \in Fo_\Sigma(V)^I$ ,  $\langle t_i \rangle_{i \in I}^{\mathcal{A}}(g) = \langle t_i^{\mathcal{A}}(g) \rangle_{i \in I}$ .
- For all  $(t_i : e_i \rightarrow e)_{i \in I} \in Fo_\Sigma(V)^I$  and  $i \in I$ ,  $\pi_i((t_i)_{i \in I}^{\mathcal{A}}(g)) = t_i^{\mathcal{A}}(g)$ .
- For all  $f : e \rightarrow e' \in Fo_\Sigma(V)$ ,  $map(f)^{\mathcal{A}}(g) = \mathcal{P}(f^{\mathcal{A}}(g))$ .
- For all  $p : e \rightarrow 2 \in Fo_\Sigma(V)$ ,  $n > 0$  and  $a_1, \dots, a_n \in A_e$ ,

$$filter(p)^{\mathcal{A}}(g)(\{a_1, \dots, a_n\}) = \{a_{i_1}, \dots, a_{i_k}\}$$

where  $\{i_1, \dots, i_k\} = \{j \in [n] \mid p^{\mathcal{A}}(g)(a_j) = 1\}$  and  $i_1 < \dots < i_k$ .

- For all  $f : e \times e' \rightarrow e \in Fo_\Sigma(V)$ ,  $a \in A_e$ ,  $b \in A_{e'}$  and  $w \in A_{e'^*}$ ,

$$foldl(f)^{\mathcal{A}}(g)(a)(\epsilon) = a,$$

$$foldl(f)^{\mathcal{A}}(g)(a)(b \cdot w) = foldl(f)^{\mathcal{A}}(g)(f^{\mathcal{A}}(g)(a, b))(w).$$

- For all  $sucs : e \rightarrow \mathcal{P}(e)$ ,  $f : e \times e' \rightarrow \mathcal{P}(e) \in Fo_\Sigma(V)$ ,  $t = foldlMS(sucs)(f)$ ,  $a \in A_e$ ,  $b \in A_{e'}$  and  $w \in A_{e'^*}$ ,

$$t^{\mathcal{A}}(g)(a)(\epsilon) = suc^{\mathcal{A}}(g)(a),$$

$$t^{\mathcal{A}}(g)(a)(b \cdot w) = t^{\mathcal{A}}(g)(joinMap^{\mathcal{A}}(curry(f)^{\mathcal{A}}(g))(suc^{\mathcal{A}}(g)(a))(b))(w).$$

- For all  $(e_i)_{i \in I} \in \mathcal{T}(S, \mathcal{I})^I$ ,  $(x_i)_{i \in I} \in \mathsf{X}_{i \in I} V_{e_i}$ ,  $t : e \in \text{Fo}_\Sigma(V)$  and  $a \in \prod_{i \in I} A_{e_i}$ ,

$$(\lambda x.t)^{\mathcal{A}}(g)(a) = t^{\mathcal{A}}(g[\pi_i(a)/x_i \mid i \in I]).$$

- For all  $f : e \rightarrow e'$ ,  $t : e \in \text{Fo}_\Sigma(V)$ ,  $f(t)^{\mathcal{A}}(g) = f^{\mathcal{A}}(g)(t^{\mathcal{A}}(g))$ .
- For all  $a \in A_e$ ,  $\text{single}^{\mathcal{A}}(a) = \{a\}$ .
- $(\text{true} : 2)^{\mathcal{A}} = 1$  and  $(\text{false} : 2)^{\mathcal{A}} = 0$ .
- For all  $(\text{true} : \mathcal{P}(e))^{\mathcal{A}} = A_e$  and  $(\text{false} : \mathcal{P}(e))^{\mathcal{A}} = \emptyset$ .
- For all  $a \in 2$ ,  $\neg^{\mathcal{A}}(a) = 1 - a$ .
- For all  $a, b \in 2$ ,  $(a \wedge^{\mathcal{A}} b) = a * b$ ,  $(a \vee^{\mathcal{A}} b) = \max(a, b)$ .
- For all  $a, b \subseteq A_e$ ,  $(a \wedge^{\mathcal{A}} b) = a \cap b$ ,  $(a \vee^{\mathcal{A}} b) = a \cup b$  and  $(a -^{\mathcal{A}} b) = a \setminus b$ .
- For all  $a, b \in A_e$ ,

$$(a =^{\mathcal{A}} b) = 1 \Leftrightarrow a = b \quad \text{and} \quad (a \neq^{\mathcal{A}} b) = 1 \Leftrightarrow a \neq b.$$

- For all  $x : e \in V$  and  $\varphi : 2 \in \text{Fo}_\Sigma(V)$ ,

$$\begin{aligned} (\forall x \varphi)^{\mathcal{A}}(g) = 1 &\Leftrightarrow \forall a \in A_e : \varphi^{\mathcal{A}}(g[a/x]) = 1, \\ (\exists x \varphi)^{\mathcal{A}}(g) = 1 &\Leftrightarrow \exists a \in A_e : \varphi^{\mathcal{A}}(g[a/x]) = 1. \end{aligned}$$

- For all  $x : e \in V$  and  $\varphi : \mathcal{P}(e') \in Fo_{\Sigma}(V)$ ,

$$\begin{aligned} (\forall x \varphi)^{\mathcal{A}}(g) &= \bigcap \{\varphi^{\mathcal{A}}(g[a/x]) \mid a \in A_e\}, \\ (\exists x \varphi)^{\mathcal{A}}(g) &= \bigcup \{\varphi^{\mathcal{A}}(g[a/x]) \mid a \in A_e\}. \end{aligned}$$

- For all  $R \subseteq A_e \times A_{e'}$  and  $a \in A_e$ ,  $\text{rel2fun}^{\mathcal{A}}(R)(a) = \{b \in A_{e'} \mid (a, b) \in R\}$ .
- For all  $p : e \rightarrow 2$  and  $r : e \times e' \rightarrow 2 \in Fo_{\Sigma}(V)$ ,

$$\begin{aligned} \text{sat}(p)^{\mathcal{A}}(g) &= \{a \in A_e \mid p^{\mathcal{A}}(g)(a) = 1\}, \\ \text{sat}(r)^{\mathcal{A}}(g) &= \{a \in A_e \times A_{e'} \mid r^{\mathcal{A}}(g)(a) = 1\}. \end{aligned}$$

- For all  $p : \mathcal{P}(e') \rightarrow 2 \in Fo_{\Sigma}(V)$  and  $R \subseteq A_e \times A_{e'}$ ,

$$\text{select}(p)^{\mathcal{A}}(g)(R) = \{a \in A_e \mid p^{\mathcal{A}}(g)(\text{rel2fun}^{\mathcal{A}}(R)(a)) = 1\}.$$

- For all  $R \subseteq A_e \times A_{e'}$ ,  $\text{inv}^{\mathcal{A}}(R) = \{(b, a) \mid (a, b) \in R\}$ .
- For all  $R \subseteq A_e \times A_{e'}$ ,

$$\begin{aligned} \pi_1^{\mathcal{A}}(R) &= \{a \in A_e \mid \exists b \in A_{e'} : (a, b) \in R\}, \\ \pi_2^{\mathcal{A}}(R) &= \{b \in A_{e'} \mid \exists a \in A_e : (a, b) \in R\}. \end{aligned}$$

- For all  $P \subseteq A_e$ ,  $P' \subseteq A_{e'}$ ,  $R \subseteq A_e \times A_{e'}$  and  $R' \subseteq A_{e'} \times A_{e''}$ ,

$$P *^{\mathcal{A}} P' = P \times P',$$

$$R /^{\mathcal{A}} P' = \{a \in A_e \mid P' \subseteq \text{rel2fun}^{\mathcal{A}}(R)(a)\},$$

$$R ;^{\mathcal{A}} R' = \{(a, c) \in \mid \exists b \in A_{e'} : (a, b) \in R \wedge (b, c) \in R'\}.$$

- For all  $R \subseteq A_e \times A_{e'}$  and  $P \subseteq A_{e'}$ ,

$$\exists^{\mathcal{A}}(R)(P) = \{a \in A_e \mid \text{rel2fun}^{\mathcal{A}}(R)(a) \cap P \neq \emptyset\},$$

$$\forall^{\mathcal{A}}(R)(P) = \{a \in A_e \mid \text{rel2fun}^{\mathcal{A}}(R)(a) \subseteq P\}.$$

- Let  $A_e$  be finite,  $x : \mathcal{P}(e) \in V$ ,  $\varphi : \mathcal{P}(e) \in \text{Fo}_{\Sigma}(V)$  and

$$\begin{aligned} \Phi : \mathcal{P}(A_e) &\rightarrow \mathcal{P}(A_e) \\ P &\mapsto \varphi^{\mathcal{A}}(g[P/x]). \end{aligned}$$

Suppose that all free occurrences of variables in  $\varphi$  have positive polarity. Then  $\Phi$  is monotone and thus by the Fixpoint Theorem for finite posets (see chapter 3),

$$(\mu x.\varphi)^{\mathcal{A}}(g) = \bigcup_{i \in \mathbb{N}} \Phi^i(\emptyset),$$

$$(\nu x.\varphi)^{\mathcal{A}}(g) = \bigcap_{i \in \mathbb{N}} \Phi^i(A_e)$$

are the least resp. greatest predicates  $P \subseteq A_e$  with  $P = \Phi(P)$ .

- Let  $A_e$  and  $A_{e'}$  be finite,  $x : \mathcal{P}(e \times e') \in V$ ,  $\rho : \mathcal{P}(e \times e') \in Fo_\Sigma(V)$  and

$$\begin{aligned}\Psi : \mathcal{P}(A_e \times A_{e'}) &\rightarrow \mathcal{P}(A_e \times A_{e'}) \\ R &\mapsto \rho^{\mathcal{A}}(g[R/x]).\end{aligned}$$

Suppose that all free occurrences of variables in  $\varphi$  have positive polarity. Then  $\Psi$  is monotone and thus by the Fixpoint Theorem for finite posets (see chapter 3),

$$\begin{aligned}(\mu x.\rho)^{\mathcal{A}}(g) &= \bigcup_{i \in \mathbb{N}} \Psi^i(\emptyset), \\ (\nu x.\rho)^{\mathcal{A}}(g) &= \bigcap_{i \in \mathbb{N}} \Psi^i(A_e \times A_{e'}).\end{aligned}$$

are the least resp. greatest relations  $R \subseteq A_e^2$  with  $R = \Psi(R)$ .

- Let  $Q$  be finite,  $x : \mathcal{P}(\text{state}^2) \in V$  and

$$\begin{aligned}\Psi' : \mathcal{P}(Q^2) &\rightarrow \mathcal{P}(Q^2) \\ R &\mapsto \bigcap_{L' \subseteq L} \ker(\text{out}^{\mathcal{A}}(L')) \\ &\quad \cap \{(q, q') \in Q^2 \mid \forall L' \subseteq L : \\ &\quad \quad \quad (\text{children}^{\mathcal{A}}(L')(q), \text{children}^{\mathcal{A}}(L')(q')) \in R_{\mathcal{P}(Q)}\}\end{aligned}$$

where  $R_{\mathcal{P}(Q)}$  is the lifting of  $R$  from  $Q$  to  $\mathcal{P}(Q)$  (see chapter 7).

$\Psi'$  is monotone and thus by the Fixpoint Theorem for finite posets (see chapter 3),

$$\sim^{\mathcal{A}} = \bigcap_{i \in \mathbb{N}} \Psi'^i(Q^2)$$

is the greatest relation  $R \subseteq Q^2$  with  $R = \Psi'(R)$ .

- $\text{labels}^{\mathcal{A}} = A_{\text{label}}$ .
- For all  $q \in Q$ ,  $\text{preds}^{\mathcal{A}}(q) = \{q' \in Q \mid q \in \text{trans}^{\mathcal{A}}(q')\}$ .
- For all  $q \in Q$  and  $lab \in L$ ,  $\text{predsL}^{\mathcal{A}}(q)(lab) = \{q' \in Q \mid q \in \text{transL}^{\mathcal{A}}(q', lab)\}$ .
- For all  $q \in Q$ ,  $\text{out}^{\mathcal{A}}(q) = \{at \in At \mid q \in \text{value}^{\mathcal{A}}(at)\}$ .
- For all  $q \in Q$  and  $lab \in L$ ,  $\text{outL}^{\mathcal{A}}(q)(lab) = \{at \in At \mid q \in \text{valueL}^{\mathcal{A}}(at, lab)\}$ .
- $\text{child}^{\mathcal{A}}(\emptyset) = \{(q, q') \in Q^2 \mid q' \in \text{trans}^{\mathcal{A}}(q)\}$ .
- For all  $L' \subseteq L \setminus \emptyset$ ,

$$\text{child}^{\mathcal{A}}(L') = \{(q, q') \in Q^2 \mid \exists lab \in L' : q' \in \text{transL}^{\mathcal{A}}(q, lab)\}.$$

- For all  $q \in Q$  and  $lab \in L$ ,

$$\text{mkrow}^{\mathcal{A}}(q)(lab) = \begin{cases} q' & \text{if } \exists q' \in Q : \text{transL}(q, lab) = \{q'\}, \\ () & \text{otherwise.} \end{cases}$$

- For all  $L' \subseteq L$ ,  $f : L \rightarrow Q + 1$  and  $lab \in L$ ,

$$\text{projectrow}^{\mathcal{A}}(L')(f)(lab) = \text{if } lab \in L' \text{ then } f(lab) \text{ else } () .$$

- For all  $p : 2 \in \text{Fo}_{\Sigma}(V)$  and  $f : L \rightarrow Q + 1$ ,

$$\text{mkinstance}(p)^{\mathcal{A}}(g)(f) = \begin{cases} 1 & \text{if } p^{\mathcal{A}}(g[f(x)/x \mid x \in L]) = 1, \\ 0 & \text{otherwise.} \end{cases}$$

- For all  $tab, tab' \subseteq (Q + 1)^L$ ,  $f \in tab$  and  $f' \in tab'$ ,

$$\text{labs}(tab) = \{lab \in L \mid \exists f \in tab : f(lab) \neq ()\},$$

$$\pi = \text{project}^{\mathcal{A}}(\text{labs}(tab) \setminus \text{labs}(tab')),$$

$$h(f, f') = 1 \Leftrightarrow_{\text{def}} \forall lab \in L : f(lab) = f'(lab) \vee f(lab) = () \vee f'(lab) = (),$$

$$tab \wedge^{\mathcal{A}} tab' = tab \cap tab',$$

$$tab \vee^{\mathcal{A}} tab' = tab \cup tab',$$

$$tab -^{\mathcal{A}} tab' = tab \setminus tab',$$

$$tab \times^{\mathcal{A}} tab' = \text{if } \text{labs}(tab) \cap \text{labs}(tab') = \emptyset \text{ then } tab \times tab' \text{ else } \emptyset,$$

$$\begin{aligned} tab /^{\mathcal{A}} tab' = & \text{if } \text{labs}(tab') \subseteq \text{labs}(tab) \wedge tab' \neq \emptyset \\ & \text{then } \pi(tab) \setminus \pi((\pi(tab) \times tab') \setminus tab) \text{ else } \emptyset, \end{aligned}$$

$$njoin^{\mathcal{A}}(tab)(tab') = \text{filter}(h)(tab \times tab').$$

**Expander2** generates the Kripke model of  $\mathcal{A}$  from finite sets of names for states, labels and atoms and **transition axioms** of the form

$$\begin{array}{ll} \varphi ==> st \rightarrow st' & \varphi ==> st \rightarrow \text{branch\$sts} \\ \varphi ==> (st, \text{lab}) \rightarrow st' & \varphi ==> (st, \text{lab}) \rightarrow \text{branch\$sts} \end{array}$$

Here  $\varphi$  is an (optional) formula (see below) and  $st, st', \text{lab}, \text{sts}$  are terms representing states (or atoms), labels and lists of states, respectively.  $\text{branch}$  transforms the list  $[t_1, \dots, t_n]$  of terms into the term  $t_1 \langle + \rangle \dots \langle + \rangle t_n$ , which is regarded as the set  $\{t_1, \dots, t_n\}$  (see [127]).

The transition axioms define functions, which interpret the  $\Sigma$ -arrows  $\text{trans}$ ,  $\text{transL}$ ,  $\text{value}$  and  $\text{valueL}$  in  $\mathcal{A}$  (see above). For accelerating the evaluation process, Expander2 works with encodings of  $\text{trans}\{L\}$  and  $\text{value}\{L\}$  as number lists: The elements of  $Q \cup L \cup At$  are represented by their respective positions in the lists of states, labels and atoms, respectively.

$Q$  is constructed stepwise: Starting out from the set  $\text{inits}^{\mathcal{A}}$  of initial states, Expander2 iteratively applies all applicable transition axioms as rewrite rules and thus builds up  $Q$  along with the functions  $\text{trans}^{\mathcal{A}}$  and  $\text{transL}^{\mathcal{A}}$ . Finally,  $\text{value}^{\mathcal{A}}$  and  $\text{valueL}^{\mathcal{A}}$  are constructed from the transition axioms with atoms on their left-hand sides.

After the Kripke model of  $\mathcal{A}$  has been defined in terms of transitions axioms (see above),  $Q$  agrees with the set  $\text{allsuccessors}(\text{inits})^{\mathcal{A}}$  (see below). A list representation of this set is obtained by simplifying the constant **states**.

Expander2 evaluates  $\Sigma$ -formulas according to their above semantics by applying simplification axioms (see below). For instance, equations of the Expander2 specification `modal` define derived  $\Sigma$ -formulas (see above) and are used to reduce formulas with powerset types to *modal normal forms*. A normal form  $t$  is then evaluated by algebraic folding that takes place whenever  $f(t)$  is simplified where  $f$  is one of the following functions:

$$\begin{aligned} eval, \ evalG &: \mathcal{P}(\text{state}) \rightarrow \mathcal{P}(\text{state}), \\ evalA &: \mathcal{P}(\text{atom}) \rightarrow \mathcal{P}(\text{atom}), \\ evalR, \ evalRG, \ evalRM &: \mathcal{P}(\text{state}^2) \rightarrow \mathcal{P}(\text{state}^2), \\ evalT, \ evalM &: \mathcal{P}_\omega(\text{row}) \rightarrow \mathcal{P}_\omega(\text{row}). \end{aligned}$$

Applying these functions to modal normal forms is more efficient than reducing the latter by ordinary simplification because *eval* et al. induce the folding in a  $\Sigma$ -algebra whose carriers consist of the *indices* of the lists of states, labels or atoms that were created when the Kripke model is generated and whose elements are often complex terms.

$trans$  and  $value$  are represented as lists of natural numbers and  $transL$  and  $valueL$  become lists of lists of natural numbers such that for all  $i, j, k \in \mathbb{N}$ ,

$$\begin{aligned} i \in \text{trans}!!j &\iff \text{states}!!i \in trans(\text{states}!!j), \\ i \in \text{transL}!!j!!k &\iff \text{states}!!i \in transL(\text{states}!!j)(\text{labels}!!k), \\ i \in \text{value}!!j &\iff \text{atoms}!!i \in value(\text{atoms}!!j), \\ i \in \text{valueL}!!j!!k &\iff \text{atoms}!!i \in valueL(\text{atoms}!!j)(\text{labels}!!k). \end{aligned}$$

For  $\varphi : \mathcal{P}(state) \in Fo_{\Sigma}(V)$ ,  $eval(\varphi)$  and  $evalG(\varphi)$  compute the subset  $\varphi^A$  of  $Q$  and display it as a list resp. graph on the canvas of Expander's solver. The graph represents  $trans^A$  and  $transL^A$  with the states of  $\varphi^A$  colored green and the states of  $Q \setminus \varphi^A$  colored red.

For  $\varphi : \mathcal{P}(atom) \in Fo_{\Sigma}(V)$ ,  $evalA(\varphi)$  computes the subset  $\varphi^A$  of  $At$  and display it as a list on the canvas of Expander's solver.

For  $\rho : \mathcal{P}(state^2) \in Fo_{\Sigma}(V)$ ,  $evalR(\rho)$  and  $evalRG(\rho)$  compute the binary relation  $\rho^A$  between states (and atoms) and display it as a list of pairs resp. the graph representing  $\rho^A$  on the canvas of Expander's solver.

*evalRM*( $\rho$ ) also computes the relation  $\rho^{\mathcal{A}}$ , but displays the matrix representing  $\rho^{\mathcal{A}}$  on the canvas of Expander's painter—after *matrices* has been selected in the interpreter menu (below the *paint* button) and the *paint* button has been pressed.

For  $\vartheta : \mathcal{P}_\omega(\text{row}) \in \text{Fo}_\Sigma(V)$ , *evalT*( $\vartheta$ ) computes the table  $\vartheta^{\mathcal{A}}$  and display it as a list of triples (row number, attribute, attribute value) on the canvas of Expander's solver. *evalM*( $\vartheta$ ) computes  $\vartheta^{\mathcal{A}}$  and displays the matrix representing  $\vartheta^{\mathcal{A}}$  on the canvas of Expander's painter—again after *matrices* has been selected in the interpreter menu and the *paint* button has been pressed.

A  $\Sigma$ -formula of the form *sat*( $p$ ) can be used for defining an atom in terms of a transition axiom (see above): Given closed  $\Sigma$ -formulas  $p : \text{state} \rightarrow 2$  and  $at : \text{atom}$ , the axiom  $at \rightarrow \text{sat}(p)$  adds all  $q \in Q$  with  $p^{\mathcal{A}}(q) = 1$  to the set *value*(*at*) $^{\mathcal{A}}$ . Since *at* and *p* are evaluated to the same set of states, the introduction of *at* seems to be expendable. However, it avoids the repeated computation of *sat*( $p$ ) $^{\mathcal{A}}$  if *p* is part of several  $\Sigma$ -formulas that are to be evaluated.

Replacing  $p$  with  $at$  in the formulas reduces the computation to the direct access to  $value(at)^{\mathcal{A}}$ . Using a label  $lab$  instead of an explicit definition of the derived function  $actL(lab)^{\mathcal{A}} : Q \rightarrow \mathcal{P}(Q)$  (see above) may have a similar optimizing effect.

Expander2 evaluates a fixpoint formula in a finite domain  $D$  with the help of following iterative function:

$$\begin{aligned} \text{fixpt} : \mathcal{P}(D \times D) &\rightarrow ((D \rightarrow D) \rightarrow (D \rightarrow D)) \\ (\leq) &\mapsto \lambda f : D \rightarrow D. \lambda d. \text{if } f(d) \leq d \text{ then } d \text{ else } \text{fixpt}(\leq)(f)(f(d)). \end{aligned}$$

In particular,

$$\begin{aligned} (\mu x. \varphi : \mathcal{P}(e))^{\mathcal{A}} &= \text{fixpt}(\subseteq)(\Phi)(\emptyset), \\ (\nu x. \varphi : \mathcal{P}(e))^{\mathcal{A}} &= \text{fixpt}(\supseteq)(\Phi)(A_e), \\ (\mu x. \rho : \mathcal{P}(e \times e'))^{\mathcal{A}} &= \text{fixpt}(\subseteq)(\Psi)(\emptyset), \\ (\nu x. \rho : \mathcal{P}(e \times e'))^{\mathcal{A}} &= \text{fixpt}(\supseteq)(\Psi)(A_e \times A_{e'}), \\ \sim^{\mathcal{A}} &= \text{fixpt}(\supseteq)(\Psi')(Q^2) \end{aligned}$$

(see above).

Expander2 implements sets by lists and the rows of a table by association lists of type  $(label \times state)^*$ .

As to types in general, Expander2 only distinguishes between **formulas**, i.e.,  $\Sigma$ -formulas of a type of the form  $e_1 \rightarrow (e_2 \rightarrow \cdots \rightarrow (e_n \rightarrow 2))$ , and **terms**, which comprise all other  $\Sigma$ -formulas. First-order formulas agree with  $\Sigma$ -formulas of type 2. Constants *true*, *false* : 2, propositional operators  $\wedge, \vee : 2^+ \rightarrow 2$  and first-order quantifiers  $\forall, \exists$  are denoted by **True**, **False**, **&**, **|**, **All** and **Any**, respectively.

Constants (like the transition relation `->`; see above) listed in the **preds** (predicates) section of an Expander2 specification are regarded as formulas, constants listed in the **constructs** (constructors) or **defuncts** (defined functions) section are regarded as terms. Further constants used in specification are supposed to be defined functions, even if they lack axioms.

Expander2 uses particular notations for first-order constants and operators because many powerful rules built into Expander's simplifier are tailored to first-order formulas and thus do not apply to synonymous higher-order constants or operators.

Besides transition axioms there are two further kinds of axioms to be used by Expander's simplifier:

$$\begin{aligned}\varphi &\implies t == t' \\ \varphi &\implies (\varphi_1 \iff \varphi_2)\end{aligned}$$

Here  $t$  and  $t'$  are terms and  $\varphi, \varphi_1, \varphi_2$  are formulas. Again,  $\varphi$  is optional. The axiom is applied to a formula  $\psi$  by matching  $t$  or  $\varphi_1$  with a subterm resp. subformula of  $\psi$  only if the respective instance of the **guard**  $\varphi$  reduces to True.

Variables are listed in the **fovars** (first-order variables) or **hovars** (higher-order variables) section of Expander2 specifications. The assignment of a variable  $x$  to the latter section is needed only if the specification contains non-leaf occurrences of  $x$ .

Two  $\Sigma$ -formulas  $\varphi$  and  $\psi$  are  **$\mathcal{A}$ -equivalent** if  $\varphi^{\mathcal{A}} = \psi^{\mathcal{A}}$ .

$g \in A^V$  solves  $\varphi : 2 \in Fo_{\Sigma}(V)$  in  $\mathcal{A}$ , written as  $\mathcal{A} \models_g \varphi$ , if  $\varphi^{\mathcal{A}}(g) = 1$ .

**Proposition VALFORM** For all  $\varphi : 2 \in Fo_{\Sigma}(V)$  and  $g, h \in A^V$ ,

$$\varphi^{\mathcal{A}}(g) = 1 \wedge g =_{free(\varphi)} h \Rightarrow \varphi^{\mathcal{A}}(h) = 1. \quad \square$$

A solution  $g$  of  $\varphi : 2 \in Fo_{\Sigma}(V)$  solves  $\varphi$  **uniquely** if for all solutions  $h$  of  $\varphi$ ,

$$h|_{free(\varphi)} = g|_{free(\varphi)}.$$

$\varphi \in Fo_{\Sigma}(V)$  is **negation-free** if neither  $\neg$  nor  $\Rightarrow$  occurs in  $\varphi$ .

## Lemma NEGFREE

For all negation-free  $\varphi : 2 \in \text{Ho}_\Sigma(V)$ ,  $g \in A^V$  and  $\Sigma$ -homomorphisms  $h : \mathcal{A} \rightarrow \mathcal{B}$ ,  $\varphi^{\mathcal{A}}(g) = 1$  implies  $\varphi^{\mathcal{B}}(h \circ g) = 1$ .  $\square$

$\mathcal{A}$  **satisfies**  $\varphi : 2 \in \text{Fo}_\Sigma(V)$ , written as  $\mathcal{A} \models \varphi$ , if for all  $g \in A^V$ ,  $\varphi^{\mathcal{A}}(g) = 1$ .

A class  $\mathcal{K}$  of  $\Sigma$ -algebras **satisfies**  $\varphi : 2 \in \text{Fo}_\Sigma(V)$  if all  $\mathcal{A} \in \mathcal{K}$  satisfy  $\varphi$ .

A class  $\mathcal{K}$  of  $\Sigma$ -algebras **satisfies**  $\Phi \subseteq \text{Fo}_\Sigma(V)_2$  if all  $\mathcal{A} \in \mathcal{K}$  satisfy all  $\varphi \in \Phi$ .

Given a set  $AX$  of  $\Sigma$ -formulas of type 2,  $\mathcal{A}$  is a  **$(\Sigma, AX)$ -algebra** if  $\mathcal{A}$  satisfies  $AX$ .

$\text{Alg}_{\Sigma, AX}$  denotes the full subcategory of  $\text{Alg}_\Sigma$  whose objects are all  $(\Sigma, AX)$ -algebras.

## Automata for satisfiability (see, e.g., [68, 162, 163])

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .

Given  $e \in \mathcal{T}(S, \mathcal{I})$ ,  $a \in A_e$  **satisfies**  $\varphi : \mathcal{P}(e) \in Fo_{\Sigma}(V)$ , written as  $\textcolor{red}{a} \models \varphi$ , if  $a \in \varphi^{\mathcal{A}}$ .

Acceptors are used for proving that a given element  $a \in A_e$  satisfies  $\varphi$ .  $\varphi$  is transformed into an automaton  $Aut(\varphi)$ , which “scans”  $a$  and achieves an accepting (or final) state if and only if  $a \in \varphi^{\mathcal{A}}$ , i.e.,

$$Aut(\varphi) \text{ accepts } a \Leftrightarrow a \in \varphi^{\mathcal{A}}. \quad (1)$$

Vice versa, an acceptor  $Aut$  of elements of  $A$  reaches a final state when scanning  $a \in A$  if and only if  $a$  belongs to the **language**  $Lan(Aut) \subseteq A$  accepted by  $Aut$ :

$$Lan(Aut) =_{def} \{a \in A \mid Aut \text{ accepts } a\}.$$

For instance, let  $\Sigma = Reg(X)$ ,  $\mathcal{A} = Lang(X)$ ,  $\varphi \in T_{Reg(X)}$  and  $Aut(\varphi)$  be the initial automaton  $(Bro(X), \varphi)$  (see sample algebras 19, 20 and 23 and sample final algebra 6).

Then  $A = \mathcal{P}(X^*)$  and  $\varphi^{\mathcal{A}} = fold^{\mathcal{A}}(\varphi) \subseteq X^*$ .

Since  $fold^{Lang(X)} = unfold^{Bro(X)}$  (see Biinductive definition of regular operators), we obtain

$$Aut(\varphi) \text{ accepts } w \in X^* \Leftrightarrow w \in unfold^{Bro(X)}(\varphi) = fold^{Lang(X)}(\varphi) = \varphi^{\mathcal{A}},$$

i.e. (1) holds true

Presumably, there are many other instances where (1) can be reduced to an equation between a fold and an unfold. Also in the following case?

**Second-order logics** involve variables for relations, *monadic* second-order logic only for unary relations, i.e., for sets or lists. In applications, these sublists of a given domain of *states*, indices of a word, nodes of a tree or graph, coordinates of a plane, etc., all called nodes in the sequel.

Since  $\Sigma$ -formulas as defined above are based on a signature that admits many sorts, we may stay with first-order logic and distinguish between variables for single nodes on the one hand and variables for sets of nodes on the other hand.

Hence MSO formulas—as defined in, e.g., [163], sections 1.1 and 3.3.2—, which express properties of words over  $X$  or finite  $C\Sigma$ -terms (see chapter 9), coincide with  $\Sigma$ -formulas as defined above where  $\Sigma$  is one of the following signatures:

$$\begin{aligned} \text{MSO}_{\text{word}}(X) &= (S, \emptyset, P), \\ S &= \{\text{node}\}, \\ F &= \{\text{label} : \text{node} \rightarrow 1 + X, \text{ succ} : \text{node} \rightarrow \text{node}\}, \\ P &= \{=, < : \text{node} \times \text{node}, \in : \text{node} \times \text{node}^*\}, \end{aligned}$$

$$\begin{aligned}
MSO_{tree}(X) &= (S, \emptyset, P), \\
S &= \{node\}, \\
F &= \{label : node \rightarrow 1 + X, \text{ children} : node \rightarrow node^*\}, \\
P &= \{=, < : node \times node, \in : node \times node^*\}.
\end{aligned}$$

## Word acceptors

Let  $w = (x_1, \dots, x_n) \in X^*$ . The  $MSO_{word}(X)$ -structure  $\underline{w}$  is defined as follows:

For all  $i \in \mathbb{N}$ ,

$$\begin{aligned}
\underline{w}_{node} &= \mathbb{N}, \\
label^{\underline{w}}(i) &= \text{if } 1 \leq i \leq |w| \text{ then } x_i \text{ else } \epsilon, \\
succ^{\underline{w}}(i) &= i + 1, \\
=^{\underline{w}} &= \Delta_{\mathbb{N}}, \\
<^{\underline{w}} &= \{(i, j) \in \mathbb{N}^2 \mid i < j\}, \\
\in^{\underline{w}} &= \{(i, v) \in \mathbb{N} \times \mathbb{N}^* \mid i \in v\}.
\end{aligned}$$

Let  $V$  be an  $S$ -sorted set of variables.  $\mathbb{N}^V$  denotes the set of **valuations of  $V$** , i.e., pairs  $(f : V_{node} \rightarrow \mathbb{N}, g : V_{nodes} \rightarrow \mathcal{P}(\mathbb{N}))$  of functions.

$(f, g) \in \mathbb{N}^V$  induces a function  $\text{vars}_{f,g} : \mathbb{N} \rightarrow \mathcal{P}(V)$  that is defined as follows: For all  $i \in \mathbb{N}$ ,

$$\text{vars}_{f,g}(i) = \{x \in V_{node} \mid f(x) = i\} \cup \{x \in V_{nodes} \mid i \in g(x)\}.$$

Let  $X_V = X \times \mathcal{P}(V)$ ,  $\mathcal{A}$  be a nondeterministic acceptor of  $X_V$ -words, i.e., an  $N\text{Acc}(X_V)$ -algebra, and  $s \in \mathcal{A}(\text{state})$ .

The language of words over  $X_V$  accepted by the initial automaton  $(\mathcal{A}, s)$  is given by  $\text{unfold}^{\mathcal{A}}(s)$  where

$$\text{unfold}^{\mathcal{A}} : \mathcal{A}(\text{state}) \rightarrow \mathcal{P}(X_V^*)$$

is the unique  $N\text{Acc}(X_V)$ -homomorphism from  $\mathcal{A}$  to the final  $N\text{Acc}(X)$ -algebra  $N\text{Pow}(X_V)$  (see section 9.1, Example 21). The acceptor of  $X_V$ -words becomes an acceptor of  $X$ -words by defining the **word language accepted by**  $(\mathcal{A}, s)$  as follows:

$$\text{Lan}(\mathcal{A}, s) = \{w \in X^* \mid \exists val \in \mathbb{N}^V : h(w, val) \in \text{unfold}^{\mathcal{A}}(s)\}$$

where

$$h : X^* \times \mathbb{N}^V \rightarrow X_V^*$$

$$((x_1, \dots, x_n), (f, g)) \mapsto ((x_1, \text{vars}_{f,g}(1)), \dots, (x_1, \text{vars}_{f,g}(1))).$$

The actual goal is to use  $(\mathcal{A}, s)$  as an automaton that accepts  $w \in X^*$  iff the  $MSO_{word}(X)$ -structure  $\underline{w}$  defined above satisfies a given  $MSO_{word}(X)$ -formula.

Indeed, by [163], Theorem 1.18, for every  $MSO_{word}(X)$ -formula  $\varphi$  over  $V$  there is an initial automaton  $(Aut(\varphi), s)$  such that for all  $w \in X^*$ ,

$$h(w, val) \in \text{unfold}^{Aut(\varphi)}(s) \Leftrightarrow val \in \varphi^w. \quad (2)$$

For the definition of  $\varphi^w$ , the set of formulas satisfied by  $w$ , see  $\Sigma$ -formulas.

If  $\varphi$  is closed, then  $\varphi^w$  is empty or equal to  $\mathbb{N}^V$ . Consequently,

$$\begin{aligned} w \in \text{Lan}(Aut(\varphi), s) &\Leftrightarrow \exists val \in \mathbb{N}^V : h(w, val) \in \text{unfold}^{Aut(\varphi)}(s) \stackrel{(2)}{\Leftrightarrow} \varphi^w \neq \emptyset \\ \varphi \text{ is closed} &\Leftrightarrow \varphi^w = \mathbb{N}^V \Leftrightarrow \underline{w} \models \varphi. \end{aligned}$$

Some formulas expressions conditions on a transition system with an arbitrary finite number of processes are expressible as  $MSO_{word}(X)$ -formulas where words over  $X$  represent states.

For instance, in [68], section 5,  $X = \{EAT, THINK, READ\}$  and  $(x_1, \dots, x_n) \in X^*$  represents the global state of a system with  $n$  processes (here: philosophers) where for all  $1 \leq i \leq n$ , the  $i$ -th process is in state  $x_i$ .

Formulas  $\varphi$  to be proved by running  $Aut(\varphi)$  come as implications whose premise describes the transition rules of the system, while the conclusion is a requirement to the system. In the example, transitions triggering actions are *eat*, *think*, *read* and *hungry*.

## Tree acceptors

Let  $\Sigma = (S, \mathcal{I}, C)$  be a finitary signature and  $t \in T_\Sigma$ . The  $MSO_{tree}(C)$ -structure  $\underline{t}$  is defined as follows: For all  $w \in \mathbb{N}^*$ ,

$$\begin{aligned}\underline{t}_{node} &= \mathbb{N}^*, \\ \underline{label}^{\underline{t}}(w) &= \text{if } w \in \text{def}(t) \text{ then } t(w) \text{ else } \epsilon, \\ \underline{children}^{\underline{t}}(w) &= [wi \mid i \in \mathbb{N}, wi \in \text{def}(t)], \\ \underline{=}^{\underline{t}} &= \Delta_{\mathbb{N}^*}, \\ \underline{<}^{\underline{t}} &= \{(v, w) \in (\mathbb{N}^*)^2 \mid \exists v' \in \mathbb{N}^+ : vv' = w\}, \\ \underline{\in}^{\underline{t}} &= \{(v, W) \in \mathbb{N}^* \times (\mathbb{N}^*)^* \mid v \in W\}.\end{aligned}$$

Let  $V$  be an  $S$ -sorted set of variables.  $(\mathbb{N}^*)^V$  denotes the set of **valuations of  $V$  in  $\mathbb{N}^*$** , i.e., pairs  $(f : V_{node} \rightarrow \mathbb{N}^*, g : V_{nodes} \rightarrow \mathcal{P}(\mathbb{N}^*))$  of functions.

$(f, g) \in (\mathbb{N}^*)^V$  induces a function  $\underline{vars}_{f,g} : \mathbb{N}^* \rightarrow \mathcal{P}(V)$  that is defined as follows: For all  $w \in \mathbb{N}^*$ ,

$$\underline{vars}_{f,g}(w) = \{x \in V_{node} \mid f(x) = w\} \cup \{x \in V_{nodes} \mid w \in g(x)\}.$$

Let  $\Sigma_V = (S, \mathcal{I}, \{(c, V') : e \rightarrow s \mid c : e \rightarrow s, V' \subseteq V\})$  and  $\mathcal{A}$  be a nondeterministic top-down acceptor of ground  $\Sigma$ -terms, i.e., an  $NTAcc(\Sigma_V)$ -algebra, and  $s \in \mathcal{A}(\text{state})$ .

The language of  $\Sigma_V$ -terms accepted by the initial automaton  $(\mathcal{A}, s)$  is given by  $unfold^{\mathcal{A}}(s)$  where

$$unfold^{\mathcal{A}} : \mathcal{A}(state) \rightarrow \mathcal{P}(T_{\Sigma_V})$$

is the unique  $NTAcc(\Sigma_V)$ -homomorphism from  $\mathcal{A}$  to the final  $NTAcc(\Sigma_V)$ -algebra  $NTPow(\Sigma_V)$  (see section 9.1, Example 30). The acceptor of  $\Sigma_V$ -terms becomes an acceptor of  $\Sigma$ -terms by defining **tree language accepted by**  $(\mathcal{A}, s)$  as follows:

$$Lan(\mathcal{A}, s) = \{t \in T_{\Sigma} \mid \exists val \in \mathbb{N}^V : h(t, val) \in unfold^{\mathcal{A}}(s)\}$$

where

$$\begin{aligned} h : T_{\Sigma} \times \mathbb{N}^V &\rightarrow T_{\Sigma_V} \\ (t, (f, g)) &\mapsto \lambda w.(t(w), vars_{f,g}(w)). \end{aligned}$$

The actual goal is to use  $(\mathcal{A}, s)$  as an automaton that accepts a  $\Sigma$ -term  $t$  iff the  $MSO_{tree}(C)$ -structure  $\underline{t}$  defined above satisfies some given  $MSO_{tree}(C)$ -formula. Indeed, by [163], Theorem 3.58, for every  $MSO_{tree}(C)$ -formula  $\varphi$  over  $V$  there is an initial automaton  $(Aut(\varphi), s)$  such that for all  $t \in T_{\Sigma}$ ,

$$h(t, val) \in unfold^{\mathcal{A}}(s) \Leftrightarrow val \in \varphi^t. \quad (3)$$

For the definition of  $\varphi^t$ , the set of formulas satisfied by  $t$ , see  $\Sigma$ -formulas.

If  $\varphi$  is closed, then  $\varphi^t$  is empty or equal to  $(\mathbb{N}^*)^V$ . Consequently,

$$t \in \text{Lan}(\text{Aut}(\varphi), s) \Leftrightarrow \exists \text{ val} \in (\mathbb{N}^*)^V : h(t, \text{val}) \in \text{unfold}^{\text{Aut}(\varphi)}(s) \stackrel{(3)}{\Leftrightarrow} \varphi^t \neq \emptyset$$

$$\varphi \text{ is closed} \Leftrightarrow \varphi^t = (\mathbb{N}^*)^V \Leftrightarrow \underline{t} \models \varphi.$$

## Institutions

An **institution** (see [54]) consists of

- a category  $Sign$  of **signatures**,
- a functor

$$\begin{aligned} Sen : Sign &\rightarrow Set \\ \Sigma &\mapsto \text{set of } \Sigma\text{-sentences} \\ \sigma : \Sigma \rightarrow \Sigma' &\mapsto Sen(\sigma) : Sen(\Sigma) \rightarrow Sen(\Sigma'), \end{aligned}$$

- a functor

$$\begin{aligned} Mod : Sign &\rightarrow Cat^{op} \\ \Sigma &\mapsto \text{category of } \Sigma\text{-models} \\ \sigma : \Sigma \rightarrow \Sigma' &\mapsto Mod(\sigma) : Mod(\Sigma') \rightarrow Mod(\Sigma), \end{aligned}$$

- for each  $\Sigma \in Sign$ , a **satisfaction relation**

$$\models_{\Sigma} \subseteq Mod(\Sigma) \times Sen(\Sigma)$$

such that for all  $Sign$ -morphisms  $\sigma : \Sigma \rightarrow \Sigma'$ ,  $\mathcal{A} \in Mod(\Sigma')$  and  $\varphi \in Sen(\Sigma)$ .

$$Mod(\sigma)(\mathcal{A}) \models_{\Sigma} \varphi \Leftrightarrow \mathcal{A} \models_{\Sigma'} Sen(\sigma)(\varphi). \quad (1)$$

Suppose that

- $\text{Sign}$  is the category of signatures and signature morphisms as defined above,
- for all signatures  $\Sigma$ ,  $\text{Sen}(\Sigma)$  is the set of first-order or modal  $\Sigma$ -formulas over a fixed set of variables,
- for all signature morphisms  $\sigma : \Sigma \rightarrow \Sigma'$  and  $\Sigma$ -formulas  $\varphi$ ,  $\text{Sen}(\sigma)$  maps  $\varphi$  to  $\sigma(\varphi)$  where  $\sigma(\varphi)$  is obtained from  $\varphi$  by replacing all arrows of  $\Sigma$  by their  $\sigma$ -images,
- for all signatures  $\Sigma$ ,  $\text{Mod}(\Sigma) = \text{Alg}_\Sigma$ ,
- for all signature morphisms  $\sigma : \Sigma \rightarrow \Sigma'$  and  $\Sigma'$ -algebras  $\mathcal{A}$ ,  $\text{Mod}(\sigma)$  maps  $\mathcal{A}$  to  $\mathcal{A}|_\sigma$  (see above),
- $\models$  is the satisfaction relation defined above.

$(\text{Sign}, \text{Sen}, \text{Mod}, \models)$  is an institution.

*Proof.* (1) amounts to:

$$\mathcal{A}|_\sigma \models_\Sigma \varphi \Leftrightarrow \mathcal{A} \models_{\Sigma'} \sigma(\varphi). \quad (2)$$

The proof of (2) is straightforward (induction on the size of  $\varphi$ ). □

## Sequent logic

Let  $\Sigma = (S, \mathcal{I}, F)$  and  $\Sigma' = (S, \mathcal{I}, F \cup P)$  be signatures and  $\mathcal{C}$  be a  $\Sigma$ -algebra such that  $P$  consists of predicates.

$Alg_{\Sigma', \mathcal{C}}$  denotes the full subcategory of  $Alg_\Sigma$  consisting of all  $\Sigma'$ -algebras  $\mathcal{A}$  with  $\mathcal{A}|_\Sigma = \mathcal{C}$ . Hence the carrier of all  $\mathcal{A} \in Alg_{\Sigma', \mathcal{C}}$  is the carrier of  $\mathcal{C}$ , in the sequel denoted by  $\mathcal{A}$ .

$Alg_{\Sigma', \mathcal{C}}$  is a complete lattice with the following partial order, least or greatest element, suprema and infima: For all  $\mathcal{A}, \mathcal{B} \in Alg_{\Sigma', \mathcal{C}}$ ,

$$\mathcal{A} \leq \mathcal{B} \iff \forall p : e \rightarrow 2 \in P, a \in A_e : p^{\mathcal{A}}(a) \leq p^{\mathcal{B}}(a).$$

For all  $\mathcal{K} \subseteq Alg_{\Sigma', \mathcal{C}}$  and  $p : e \rightarrow 2 \in P$ ,

$$\begin{aligned} p^\perp &= \lambda a \in A_e. 0, & p^\top &= \lambda a \in A_e. 1, \\ p^{\sqcup \mathcal{K}} &= \lambda a \in A_e. \exists \mathcal{A} \in \mathcal{K} : p^{\mathcal{A}}(a) = 1, & p^{\sqcap \mathcal{K}} &= \lambda a \in A_e. \forall \mathcal{A} \in \mathcal{K} : p^{\mathcal{A}}(a) = 1. \end{aligned}$$

Let  $\varphi$  be a negation-free  $\Sigma'$ -formula. Then the semantics of  $\varphi$  is monotone w.r.t. the above partial order on  $Alg_{\Sigma', \mathcal{C}}$ , i.e., for all  $\mathcal{A}, \mathcal{B} \in Alg_{\Sigma', \mathcal{C}}$ ,

$$\mathcal{A} \leq \mathcal{B} \text{ implies } \varphi^{\mathcal{A}} \subseteq \varphi^{\mathcal{B}}. \quad (1)$$

(1) can be shown by induction on the size of  $\varphi$ .

A  $\Sigma'$ -formula of the form  $\varphi \Rightarrow \psi$  is called a  **$\Sigma'$ -sequent** (or **Gentzen formula**) if

- $\varphi \neq \text{False}$  and  $\varphi$  is negation-free,
- $\psi \neq \text{True}$  and  $\psi$  is negation-free.

$\varphi$  may be restricted to a conjunction and  $\psi$  to a disjunction of formulas because other  $\Sigma$ -sequents are equivalent (w.r.t.  $\text{Alg}_{\Sigma'}$ ) to conjunctions of such sequents.

If  $\varphi = \text{True}$ , then  $\varphi \Rightarrow \psi$  is often abbreviated to  $\psi$ .

If  $\psi = \text{False}$ , then  $\varphi \Rightarrow \psi$  is often abbreviated to  $\neg\varphi$ .

A **Horn clause for**  $p \in P$  is a  $\Sigma'$ -sequent of the form  $\varphi \Rightarrow p(t)$ , mostly written as  $p(t) \Leftarrow \varphi$ .

A **Horn clause for**  $f \in F$  is a  $\Sigma'$ -sequent of the form  $\varphi \Rightarrow f(t) = u$ , mostly written as  $f(t) = u \Leftarrow \varphi$ .

A **Horn clause for** a binary “transition” predicate  $\rightarrow \in F$  is a  $\Sigma'$ -sequent of the form  $\varphi \Rightarrow t \rightarrow u$ , mostly written as  $t \rightarrow u \Leftarrow \varphi$ .

In the rules given above, the premise  $\varphi$  of a Horn clause  $\psi \Leftarrow \varphi$  is sometimes splitted into a **guard**  $\gamma$  to be proved before the rule is applied and the rest  $\varphi'$  an instance of which is part of the rule reduct.  $\psi \Leftarrow \varphi$  is then written as  $\gamma \Rightarrow (\psi \Leftarrow \varphi')$ .

A **co-Horn clause for**  $p \in P$  is a  $\Sigma'$ -sequent of the form  $p(t) \Rightarrow \varphi$ .

### Example 1 (Horn and co-Horn clauses for predicates on lists)

Let  $\Sigma = \text{List}(X) \cup (\emptyset, 2, \{\leq, > : X^2 \rightarrow 2\})$ ,  $P = \{\text{sorted}, \text{unsorted} : \text{state} \rightarrow 2\}$  and  $V = \{x, y : X, s : \text{state}\}$ . Here are Horn clauses for *sorted* and *unsorted*:

$$\text{sorted}(\alpha)$$

$$\text{sorted}(\text{cons}(x, \alpha))$$

$$\text{sorted}(\text{cons}(x, \text{cons}(y, s))) \Leftarrow x \leq y \wedge \text{sorted}(\text{cons}(y, s))$$

$$\text{unsorted}(\text{cons}(x, \text{cons}(y, s))) \Leftarrow x > y$$

$$\text{unsorted}(\text{cons}(x, \text{cons}(y, s))) \Leftarrow \text{unsorted}(\text{cons}(y, s))$$

Here are co-Horn clauses for *sorted* and *unsorted*:

$$\text{sorted}(\text{cons}(x, \text{cons}(y, s))) \Rightarrow x \leq y$$

$$\text{sorted}(\text{cons}(x, \text{cons}(y, s))) \Rightarrow \text{sorted}(\text{cons}(y, s))$$

$$\neg \text{unsorted}(\alpha)$$

$$\neg \text{unsorted}(\text{cons}(x, \alpha))$$

$$\text{unsorted}(\text{cons}(x, \text{cons}(y, s))) \Rightarrow x > y \vee \text{unsorted}(\text{cons}(y, s)) \quad \square$$

**Example 2** (Horn and co-Horn clauses for predicates on streams)

Let  $\Sigma = \text{Stream}(X) \cup (\emptyset, 2, \{\leq, >: X^2 \rightarrow 2\})$ ,  $P = \{\text{sorted}, \text{unsorted} : \text{state} \rightarrow 2\}$  and  $V = \{s : \text{state}\}$ . Here are Horn clauses for *sorted* and *unsorted*:

$$\text{sorted}(s) \Leftarrow \text{head}(s) \leq \text{head}(\text{tail}(s)) \wedge \text{sorted}(\text{tail}(s))$$

$$\text{unsorted}(s) \Leftarrow \text{head}(s) > \text{head}(\text{tail}(s))$$

$$\text{unsorted}(s) \Leftarrow \text{unsorted}(\text{tail}(s))$$

Here are co-Horn clauses for *sorted* and *unsorted*:

$$\text{sorted}(s) \Rightarrow \text{head}(s) \leq \text{head}(\text{tail}(s))$$

$$\text{sorted}(s) \Rightarrow \text{sorted}(\text{tail}(s))$$

$$\text{unsorted}(s) \Rightarrow \text{head}(s) > \text{head}(\text{tail}(s)) \vee \text{unsorted}(\text{tail}(s)) \quad \square$$

Let  $AX$  be a set of  $\Sigma'$ -sequents and  $SP = (\Sigma, P, AX)$ . The class of all  $\Sigma'$ -algebras  $\mathcal{A} \in Alg_{\Sigma',C}$  such that  $\mathcal{A}$  satisfies all  $\Sigma$ -sequents of  $AX$ , is denoted by  $Alg_{SP,C}$ .

## Least predicates

$SP$  is a **Horn specification** if for all  $\varphi \in AX$ ,  $\varphi$  is a  $\Sigma$ -sequent or a Horn clause for some  $p \in P$ . Then  $p$  is called a **least predicate**.

The **step function** or **consequence operator**

$$\Phi = \Phi_{SP,C} : Alg_{\Sigma',C} \rightarrow Alg_{\Sigma',C}$$

is defined as follows:

For all  $\mathcal{A} \in Alg_{\Sigma',C}$ ,  $p : e \rightarrow 2 \in P$  and  $a \in A_e$ ,

$$p^{\Phi(\mathcal{A})}(a) = 1 \Leftrightarrow \exists p(t) \Leftarrow \varphi \in AX, g \in \varphi^{\mathcal{A}} : g^*(t) = a. \quad (2)$$

By (1),  $\Phi$  is monotone and thus by the Fixpoint Theorem of Knaster and Tarski,  $\Phi$  has the least fixpoint

$$lfp(\Phi) = \bigcap \{\mathcal{A} \in Alg_{\Sigma',C} \mid \Phi(\mathcal{A}) \leq \mathcal{A}\}.$$

Let  $p : e \rightarrow 2 \in P$ ,  $\text{AX}_p$  be the set of Horn clauses of  $AX$  for  $p$  and  $x \in V_e \setminus \text{var}(AX_p)_e$ . The  $\Sigma'$ -formula

$$[\text{AX}_p] =_{\text{def}} p(x) \Leftrightarrow \bigvee_{cl=(p(t) \Leftarrow \varphi) \in AX} \exists \text{ var}(cl) : (x = t \wedge \varphi)$$

is called the **Horn completion of  $AX_p$** .

The Horn completion of  $AX_p$  combines all Horn clauses of  $AX$  for  $p$  to a single one, which is actually an *equivalence* in  $\text{lfp}(\Phi)$ —just like, for a constructive signature  $\Sigma = (S, \mathcal{I}, C)$  and a sort  $s \in S$ , the sum extension

$$[c]_{c:e \rightarrow s \in C} : \coprod_{c:e \rightarrow s \in C} \rightarrow s$$

combines all arrows of  $C$  with target  $s$  to a single one whereby every  $\Sigma$ -algebra  $\mathcal{A}$  becomes an  $H_\Sigma$ -algebra, which is an *isomorphism* if  $\mathcal{A}$  is initial in  $\text{Alg}_\Sigma$  (see chapter 13). Another expression of the Curry-Howard correspondence?

## Lemma IND

$$Alg_{SP,\mathcal{C}} = \{\mathcal{A} \in Alg_{\Sigma',\mathcal{C}} \mid \Phi(\mathcal{A}) \leq \mathcal{A}\}. \quad (3)$$

Moreover, for all  $\mathcal{A} \in Alg_{SP,\mathcal{C}}$ ,

$$lfp(\Phi) \leq \mathcal{A}, \quad (4)$$

$$\Phi(\mathcal{A}) = \mathcal{A} \quad \text{iff} \quad \forall p \in P : \mathcal{A} \models [AX_p]. \quad (5)$$

*Proof.* (3) Let  $\mathcal{A} \in Alg_{SP,\mathcal{C}}$ ,  $p : e \rightarrow 2 \in P$  and  $a \in A_e$  such that  $p^{\Phi(\mathcal{A})}(a) = 1$ . Then  $a = g^*(t)$  for some  $\varphi \Rightarrow p(t) \in AX$  and  $g \in \varphi^{\mathcal{A}}$ . Since  $\mathcal{A}$  satisfies  $\varphi \Rightarrow p(t)$ ,  $g \in p(t)^{\mathcal{A}}$  and thus  $p^{\mathcal{A}}(a) = p^{\mathcal{A}}(g^*(t)) = 1$ . Hence  $p^{\Phi(\mathcal{A})} \leq p^{\mathcal{A}}$  and thus  $\Phi(\mathcal{A}) \leq \mathcal{A}$ .

Conversely, let  $\Phi(\mathcal{A}) \leq \mathcal{A}$ ,  $\varphi \Rightarrow p(t) \in AX$  and  $g \in \varphi^{\mathcal{A}}$ . Then  $p^{\Phi(\mathcal{A})}(g^*(t)) = 1$ . Since  $\Phi(\mathcal{A}) \leq \mathcal{A}$ ,  $\in p^{\mathcal{A}}(g^*(t)) = 1$  and thus  $g \in p(t)^{\mathcal{A}}$ . Therefore,  $\mathcal{A}$  satisfies  $\varphi \Rightarrow p(t)$ .

(4) For all  $\mathcal{A} \in Alg_{SP,\mathcal{C}}$  and  $p : e \rightarrow 2 \in P$ ,

$$\begin{aligned} p^{lfp(\Phi)} &= \lambda a \in A_e. \forall \mathcal{B} \in Alg_{\Sigma',\mathcal{C}} : (\Phi(\mathcal{B}) \leq \mathcal{B} \Rightarrow p^{\mathcal{B}}(a) = 1) \\ &\stackrel{(3)}{=} \lambda a \in A_e. \forall \mathcal{B} \in Alg_{SP,\mathcal{C}} : p^{\mathcal{B}}(a) = 1 \leq p^{\mathcal{A}}, \end{aligned}$$

i.e.,  $lfp(\Phi) \leq \mathcal{A}$ .

(5) Let  $\mathcal{A} \in Alg_{SP,\mathcal{C}}$  and for all  $p : e \rightarrow 2 \in P$ , let  $x_p \in V_e \setminus var(AX_p)_e$ . Then

$$\begin{aligned}
& \Phi(\mathcal{A}) = \mathcal{A} \Leftrightarrow \forall p \in P : p^{\mathcal{A}} = p^{\Phi(\mathcal{A})} \\
\Leftrightarrow & \forall p : e \rightarrow 2 \in P, a \in A_e : \\
& p^{\mathcal{A}}(a) = 1 \Leftrightarrow \exists p(t) \Leftarrow \varphi \in AX, g \in \varphi^{\mathcal{A}} : g^*(t) = a \\
\Leftrightarrow & \forall p \in P, g \in A^V : \\
& p^{\mathcal{A}}(g(x_p)) = 1 \Leftrightarrow \exists p(t) \Leftarrow \varphi \in AX, h \in \varphi^{\mathcal{A}} : h^*(t) = g(x_p) \\
\Leftrightarrow & \forall p \in P, g \in A^V : \\
& g \in p(x_p)^{\mathcal{A}} \Leftrightarrow \exists cl = (p(t) \Leftarrow \varphi) \in AX, h \in A^V : \\
& \quad g =_{var(cl)} h \wedge h(x_p) = h^*(t) \wedge h \in \varphi^{\mathcal{A}} \\
\Leftrightarrow & \forall p \in P, g \in A^V : \\
& g \in p(x_p)^{\mathcal{A}} \Leftrightarrow \exists cl = (p(t) \Leftarrow \varphi) \in AX, h \in A^V : \\
& \quad g =_{var(cl)} h \wedge h \in (x_p = t \wedge \varphi)^{\mathcal{A}} \\
\Leftrightarrow & \forall p \in P, g \in A^V : \\
& g \in p(x_p)^{\mathcal{A}} \Leftrightarrow \exists cl = (p(t) \Leftarrow \varphi) \in AX : g \in (\exists var(cl) : (x_p = t \wedge \varphi))^{\mathcal{A}} \\
\Leftrightarrow & \forall p \in P, g \in A^V : \\
& g \in p(x_p)^{\mathcal{A}} \Leftrightarrow g \in (\bigvee_{cl=(p(t) \Leftarrow \varphi) \in AX} \exists var(cl) : (x_p = t \wedge \varphi))^{\mathcal{A}} \\
\Leftrightarrow & \forall p \in P : (p(x_p) \Leftrightarrow \bigvee_{cl=(p(t) \Leftarrow \varphi) \in AX} \exists var(cl) : (x_p = t \wedge \varphi))^{\mathcal{A}} = A^V \\
\Leftrightarrow & \forall p \in P : \mathcal{A} \models [AX_p]. \quad \square
\end{aligned}$$

## Greatest predicates

$SP = (\Sigma, P, AX)$  is a **co-Horn specification** if for all  $\varphi \in AX$ ,  $\varphi$  is a  $\Sigma$ -sequent or a co-Horn clause for some  $p \in P$ . Then  $p$  is called a **greatest predicate**.

The **step function** or **consequence operator**

$$\Phi = \Phi_{SP,C} : Alg_{\Sigma',C} \rightarrow Alg_{\Sigma',C}$$

is defined as follows:

For all  $\mathcal{A} \in Alg_{\Sigma',C}$ ,  $p : e \rightarrow 2 \in P$  and  $a \in A_e$ ,

$$p^{\Phi(\mathcal{A})}(a) = 1 \Leftrightarrow \forall p(t) \Rightarrow \varphi \in AX, g \in A^V : g^*(t) = a \Rightarrow g \in \varphi^A \quad (6)$$

where  $A$  is the carrier of  $\mathcal{A}$ .

By (1),  $\Phi$  is monotone and thus by the Fixpoint Theorem of Knaster and Tarski,  $\Phi$  has the greatest fixpoint

$$gfp(\Phi) = \bigsqcup \{\mathcal{A} \in Alg_{\Sigma',C} \mid \mathcal{A} \leq \Phi(\mathcal{A})\}.$$

Let  $p : e \rightarrow 2 \in P$ ,  $AX_p$  be the set of co-Horn clauses of  $AX$  for  $p$  and  $x \in V_e \setminus var(AX_p)_e$ . The  $\Sigma'$ -formula

$$\langle AX_p \rangle =_{def} p(x) \Leftrightarrow \bigwedge_{cl=(p(t) \Rightarrow \varphi) \in AX} \forall var(cl) : (x = t \Rightarrow \varphi)$$

is called the **co-Horn completion** of  $AX_p$ .

The co-Horn completion of  $AX_p$  combines all co-Horn clauses of  $AX$  for  $p$  to a single one, which is actually an *equivalence* in  $\text{lfp}(\Phi)$ —just like, for a destructive signature  $\Sigma = (S, \mathcal{I}, D)$  and a sort  $s \in S$ , the product extension

$$[d]_{d:s \rightarrow e \in D} : s \rightarrow \prod_{d:s \rightarrow e \in D}$$

combines all arrows of  $D$  with source  $s$  to a single one whereby every  $\Sigma$ -algebra  $\mathcal{A}$  becomes an  $H_\Sigma$ -algebra, which is an *isomorphism* if  $\mathcal{A}$  is final in  $\text{Alg}_\Sigma$  (see chapter 13).

## Lemma COIND

$$\text{Alg}_{SP,\mathcal{C}} = \{\mathcal{A} \in \text{Alg}_{\Sigma',\mathcal{C}} \mid \mathcal{A} \leq \Phi(\mathcal{A})\}. \quad (7)$$

Moreover, for all  $\mathcal{A} \in \text{Alg}_{SP,\mathcal{C}}$ ,

$$\mathcal{A} \leq \text{gfp}(\Phi), \quad (8)$$

$$\Phi(\mathcal{A}) = \mathcal{A} \quad \text{iff} \quad \forall p \in P : \mathcal{A} \models \langle AX_p \rangle. \quad (9)$$

*Proof.* (7) Let  $\mathcal{A} \in \text{Alg}_{SP,\mathcal{C}}$ ,  $p : e \rightarrow 2 \in P$  and  $a \in A_e$  such that  $p^{\Phi(\mathcal{A})}(a) = 0$ . Then  $a = g^*(t)$  for some  $p(t) \Rightarrow \varphi \in AX$  and  $g \in A^V \setminus \varphi^\mathcal{A}$ .

Since  $\mathcal{A}$  satisfies  $p(t) \Rightarrow \varphi$ ,  $g \in A^V \setminus p(t)^{\mathcal{A}}$  and thus  $p^{\mathcal{A}}(a) = p^{\mathcal{A}}(g^*(t)) = 0$ . Hence  $p^{\mathcal{A}} \leq p^{\Phi(\mathcal{A})}$  and thus  $\mathcal{A} \leq \Phi(\mathcal{A})$ .

Conversely, let  $\mathcal{A} \leq \Phi(\mathcal{A})$ ,  $p(t) \Rightarrow \varphi \in AX$  and  $g \in A^V \setminus \varphi^{\mathcal{A}}$ . Then  $p^{\Phi(\mathcal{A})}(g^*(t)) = 0$ . Since  $\mathcal{A} \leq \Phi(\mathcal{A})$ ,  $p^{\mathcal{A}}(g^*(t)) = 0$  and thus  $g \notin p(t)^{\mathcal{A}}$ . Therefore,  $\mathcal{A}$  satisfies  $p(t) \Rightarrow \varphi$ .

(8) For all  $\mathcal{A} \in Alg_{SP,C}$  and  $p : e \rightarrow 2 \in P$ ,

$$\begin{aligned} p^{\mathcal{A}} &\leq \lambda a \in A_e. \exists \mathcal{B} \in Alg_{SP,C} : p^{\mathcal{B}}(a) = 1 \\ &\stackrel{(7)}{=} \lambda a \in A_e. \exists \mathcal{B} \in Alg_{\Sigma',C} : (\mathcal{B} \leq \Phi(\mathcal{B}) \wedge p^{\mathcal{B}}(a) = 1) = p^{gfp(\Phi)}, \end{aligned}$$

i.e.,  $\mathcal{A} \leq gfp(\Phi)$ .

(9) Let  $\mathcal{A} \in Alg_{SP,C}$ ,  $p : e \rightarrow 2 \in P$  and for all  $p : e \rightarrow 2 \in P$ , let  $x_p \in V_e \setminus var(AX_p)_e$ . Then

$$\begin{aligned} \Phi(\mathcal{A}) = \mathcal{A} &\Leftrightarrow \forall p \in P : p^{\mathcal{A}} = p^{\Phi(\mathcal{A})} \\ &\Leftrightarrow \forall p : e \rightarrow 2 \in P, a \in A_e : \\ &\quad p^{\mathcal{A}}(a) = 1 \Leftrightarrow \forall p(t) \Rightarrow \varphi \in AX : g^*(t) = a \Rightarrow g \in \varphi^{\mathcal{A}} \\ &\Leftrightarrow \forall p \in P, g \in A^V : \\ &\quad p^{\mathcal{A}}(g(x_p)) = 1 \Leftrightarrow \forall p(t) \Rightarrow \varphi \in AX, h \in A^V : h^*(t) = g(x_p) \Rightarrow h \in \varphi^{\mathcal{A}} \end{aligned}$$

$\Leftrightarrow \forall p \in P, g \in A^V :$

$$\begin{aligned} g \in p(x_p)^{\mathcal{A}} &\Leftrightarrow \forall cl = (p(t) \Rightarrow \varphi) \in AX, h \in A^V : \\ &g =_{var(cl)} h \wedge h(x_p) = h^*(t) \Rightarrow h \in \varphi^{\mathcal{A}} \end{aligned}$$

$\Leftrightarrow \forall p \in P, g \in A^V :$

$$\begin{aligned} g \in p(x_p)^{\mathcal{A}} &\Leftrightarrow \forall cl = (p(t) \Rightarrow \varphi) \in AX, h \in A^V : \\ &g =_{var(cl)} h \Rightarrow h \in (x_p = t \Rightarrow \varphi)^{\mathcal{A}} \end{aligned}$$

$\Leftrightarrow \forall p \in P, g \in A^V :$

$$g \in p(x_p)^{\mathcal{A}} \Leftrightarrow \forall cl = (p(t) \Rightarrow \varphi) \in AX : g \in (\forall var(cl) : (x_p = t \Rightarrow \varphi))^{\mathcal{A}}$$

$\Leftrightarrow \forall p \in P, g \in A^V :$

$$g \in p(x_p)^{\mathcal{A}} \Leftrightarrow g \in (\bigwedge_{cl=(p(t) \Rightarrow \varphi) \in AX} \forall var(cl) : (x_p = t \Rightarrow \varphi))^{\mathcal{A}}$$

$$\Leftrightarrow \forall p \in P : (p(x_p) \Leftrightarrow \bigwedge_{cl=(p(t) \Rightarrow \varphi) \in AX} \forall var(cl) : (x_p = t \Rightarrow \varphi))^{\mathcal{A}} = A^V$$

$$\Leftrightarrow \forall p \in P : \mathcal{A} \models \langle AX_p \rangle.$$

□

## Continuity of step functions and consequences

A  $\Sigma'$ -formula  $\forall x\varphi$  or  $\exists x\varphi$  is **finitely quantified** if for all  $\mathcal{A} \in Alg_{\Sigma',C}$  and  $g \in A^V$ , the set

$$\mathcal{S}(\varphi, \mathcal{A}, g) = \{h \in A^V \mid g =_{var(\varphi)} h, h \in \varphi^{\mathcal{A}}\}$$

is finite.

## Lemma FQ

Let  $\varphi$  be a negation-free  $\Sigma'$ -formula such that all subformulas  $\forall x\psi$  and  $\exists x\psi$  of  $\varphi$  are finitely quantified.

- (i) For all  $\omega$ -chains  $\{\mathcal{A}_i \mid i < \omega\}$  of  $Alg_{\Sigma',C}$ ,  $\varphi \sqcup_{i<\omega} \mathcal{A}_i \subseteq \bigcup_{i<\omega} \varphi^{\mathcal{A}_i}$ .
- (ii) For all  $\omega$ -cochains  $\{\mathcal{A}_i \mid i < \omega\}$  of  $Alg_{\Sigma',C}$ ,  $\bigcap_{i<\omega} \varphi^{\mathcal{A}_i} \subseteq \varphi \sqcap_{i<\omega} \mathcal{A}_i$ .

*Proof of (i) by induction on the size of  $\varphi$ .*

For all  $\Sigma'$ -atoms  $p(t)$ , \*\*\*

$$p(t) \sqcup_{i<\omega} \mathcal{A}_i = \{g \in C^V \mid g^*(t) \in p \sqcup_{i<\omega} \mathcal{A}_i\} = \bigcup_{i<\omega} \{g \in C^V \mid g^*(t) \in p^{\mathcal{A}_i}\} = \bigcup_{i<\omega} p(t)^{\mathcal{A}_i}.$$

Let  $\varphi, \psi \in Fo_{\Sigma'}(V)$ .

$$\begin{aligned} (\varphi \vee \psi) \sqcup_{i<\omega} \mathcal{A}_i &= \varphi \sqcup_{i<\omega} \mathcal{A}_i \cup \psi \sqcup_{i<\omega} \mathcal{A}_i \stackrel{i.h.}{\subseteq} (\bigcup_{i<\omega} \varphi^{\mathcal{A}_i}) \cup (\bigcup_{i<\omega} \psi^{\mathcal{A}_i}) = \bigcup_{i<\omega} (\varphi^{\mathcal{A}_i} \cup \psi^{\mathcal{A}_i}) \\ &= \bigcup_{i<\omega} (\varphi \vee \psi)^{\mathcal{A}_i}. \end{aligned}$$

$$\begin{aligned} (\varphi \wedge \psi) \sqcup_{i<\omega} \mathcal{A}_i &= \varphi \sqcup_{i<\omega} \mathcal{A}_i \cap \psi \sqcup_{i<\omega} \mathcal{A}_i \stackrel{i.h.}{\subseteq} (\bigcup_{i<\omega} \varphi^{\mathcal{A}_i}) \cap (\bigcup_{i<\omega} \psi^{\mathcal{A}_i}) = \bigcup_{i,j \in \mathbb{N}} (\varphi^{\mathcal{A}_i} \cap \psi^{\mathcal{A}_j}) \\ &\subseteq \bigcup_{i,j \in \mathbb{N}} (\varphi^{A_{max(i,j)}} \cap \psi^{A_{max(i,j)}}) = \bigcup_{i<\omega} (\varphi^{\mathcal{A}_i} \cap \psi^{\mathcal{A}_i}) = \bigcup_{i<\omega} (\varphi \wedge \psi)^{\mathcal{A}_i}. \end{aligned}$$

Let  $\varphi \in Fo_{\Sigma'}(V)$ ,  $s \in S$  and  $x \in V_s$ .

$$\begin{aligned} (\exists x\varphi) \sqcup_{i<\omega} A_i &= \{g \in C^V \mid \exists a \in C_s : g[a/x] \in \varphi \sqcup_{i<\omega} A_i\} \\ &\stackrel{i.h.}{\subseteq} \{g \in C^V \mid \exists a \in C_s : g[a/x] \in \bigcup_{i<\omega} \varphi^{A_i}\} \\ &= \bigcup_{i<\omega} \{g \in C^V \mid \exists a \in C_s : g[a/x] \in \varphi^{A_i}\} = \bigcup_{i<\omega} (\exists x\varphi)^{A_i}. \end{aligned}$$

Let  $g \in C^V$  and  $B_g =_{def} \{a \in C_s \mid g[a/x] \in \varphi \sqcup_{i<\omega} A_i\}$ . Suppose that

$$\exists n \in \mathbb{N} : \forall a \in B_g : g[a/x] \in \varphi^{A_n} \tag{6}$$

holds true. Then

$$\begin{aligned} (\forall x\varphi) \sqcup_{i<\omega} A_i &= \{g \in C^V \mid \forall a \in C_s : g[a/x] \in \varphi \sqcup_{i<\omega} A_i\} = \{g \in C^V \mid \forall a \in C_s : a \in B_g\} \\ &\stackrel{(6)}{=} \{g \in C^V \mid \forall a \in C_s : g[a/x] \in \varphi^{A_n}\} = (\forall x\varphi)^{A_n} \subseteq \bigcup_{i<\omega} (\forall x\varphi)^{A_i}. \end{aligned}$$

It remains to show (6). Since  $\forall x\varphi$  is finitely quantified,  $B_g$  is finite. By induction hypothesis,  $\varphi \sqcup_{i<\omega} A_i \subseteq \bigcup_{i<\omega} \varphi^{A_i}$ . Hence for all  $a \in B_g$  there is  $i_a \in \mathbb{N}$  with  $g[a/x] \in \varphi^{A_{i_a}}$ .

Since  $B_g$  is finite,  $n =_{def} \max\{i_a \mid a \in B_g\} < \omega$ . Since for all  $i < j \in \mathbb{N}$ ,  $\varphi^{A_i} \subseteq \varphi^{A_j}$ , we conclude (6).

(ii) can be shown analogously. □

**Theorem CONSTEP** Let  $SP = (\Sigma, P, AX)$  be a Horn specification. Then  $\Phi = \Phi_{SP,C}$  is  $\omega$ -continuous.

*Proof.* Let  $\{A_i \mid i < \omega\}$  be an  $\omega$ -chain of  $Alg_{\Sigma',C}$ . Since  $\Phi$  is monotone, it remains to show:

$$\Phi(\bigsqcup_{i<\omega} A_i) \leq \bigsqcup_{i<\omega} \Phi(A_i). \quad (7)$$

Let  $p \in P$ . Then by Lemma FQ (i),

$$\begin{aligned} p^{\Phi(\bigsqcup_{i<\omega} A_i)} &= \{g^*(t) \mid p(t) \Leftarrow \varphi \in AX, g \in \varphi^{\bigsqcup_{i<\omega} A_i}\} \\ &\subseteq \{g^*(t) \mid p(t) \Leftarrow \varphi \in AX, g \in \bigcup_{i<\omega} \varphi^{A_i}\} = \bigcup_{i<\omega} \{g^*(t) \mid p(t) \Leftarrow \varphi \in AX, g \in \varphi^{A_i}\} \\ &= \bigcup_{i<\omega} p^{\Phi(A_i)} = p^{\bigsqcup_{i<\omega} \Phi(A_i)}. \end{aligned}$$

Hence (7) holds true. □

## Theorem COCONSTEP

Let  $SP = (\Sigma, P, AX)$  be a co-Horn specification. Then  $\Phi = \Phi_{SP, \mathcal{C}}$  is  $\omega$ -cocontinuous.

*Proof.* Let  $\{A_i \mid i < \omega\}$  be an  $\omega$ -cochain of  $Alg_{\Sigma', \mathcal{C}}$ . Since  $\Phi$  is monotone, it remains to show:

$$\prod_{i<\omega} \Phi(A_i) \leq \Phi(\prod_{i<\omega} A_i). \quad (8)$$

Let  $p : e \in P$ . Then by Lemma FQ (ii),  $C^V \setminus \varphi^{\prod_{i<\omega} A_i} \subseteq C^V \setminus \bigcap_{i<\omega} \varphi^{A_i}$ , and thus

$$\begin{aligned} p^{\prod_{i<\omega} \Phi(A_i)} &= \bigcap_{i<\omega} p^{\Phi(A_i)} = \bigcap_{i<\omega} (C_e \setminus \{g^*(t) \mid p(t) \Rightarrow \varphi \in AX, g \in C^V \setminus \varphi^{A_i}\}) \\ &= C_e \setminus \bigcup_{i<\omega} \{g^*(t) \mid p(t) \Rightarrow \varphi \in AX, g \in C^V \setminus \varphi^{A_i}\} \\ &= C_e \setminus \{g^*(t) \mid p(t) \Rightarrow \varphi \in AX, g \in \bigcup_{i<\omega} (C^V \setminus \varphi^{A_i})\} \\ &= C_e \setminus \{g^*(t) \mid p(t) \Rightarrow \varphi \in AX, g \in C^V \setminus \bigcap_{i<\omega} \varphi^{A_i}\} \\ &\subseteq C_e \setminus \{g^*(t) \mid p(t) \Rightarrow \varphi \in AX, g \in C^V \setminus \varphi^{\prod_{i<\omega} A_i}\} = p^{\Phi(\prod_{i<\omega} A_i)}. \end{aligned}$$

Hence (8) holds true. □

## Lemma MUPRED

Let  $SP = (\Sigma, P, AX)$  be a Horn specification and  $\mathcal{A}$  be a  $\Sigma'$ -algebra with carrier  $A$  that satisfies  $AX$ .

Every  $\Sigma$ -homomorphism  $h : \mathcal{C} \rightarrow \mathcal{A}|_\Sigma$  is a  $\Sigma'$ -homomorphism from  $lfp(\Phi)$  to  $\mathcal{A}$ .

In particular, if  $\mathcal{C}$  is initial in  $Alg_\Sigma$ , then  $lfp(\Phi)$  is initial in  $Alg_{\Sigma',AX}$ .

*Proof.* It is sufficient to show that for all  $p \in P$ ,

$$h(p^{lfp(\Phi)}) \subseteq p^{\mathcal{A}},$$

or, equivalently, by Lemma CONSTEP and Kleene's Fixpoint Theorem (1), for all  $i \in \mathbb{N}$ ,

$$h(p^{\Phi^i(\perp)}) \subseteq p^{\mathcal{A}}. \quad (9)$$

*Case 1:*  $i = 0$ . Since  $p^\perp = \emptyset$ , (9) holds true trivially.

*Case 2:* Let  $i > 0$  and  $c \in p^{\Phi^i(\perp)}$ . Then  $c = g^*(t)$  for some  $\varphi \Rightarrow p(t) \in AX$  and  $g \in \varphi^{\Phi^{i-1}(\perp)}$ . By induction hypothesis,  $h$  is a  $\Sigma'$ -homomorphism from  $\Phi^{i-1}(\perp)$  to  $\mathcal{A}$ . Hence by Lemma NEGFREE,  $h \circ g \in \varphi^{\mathcal{A}}$ . Since  $\mathcal{A}$  satisfies  $p(t) \Leftarrow \varphi$ , we conclude  $h \circ g \in p(t)^{\mathcal{A}} = \{f \in A^V \mid f^*(t) \in p^{\mathcal{A}}\}$  and thus

$$h(c) = h(g^*(t)) = (h \circ g)^*(t) \in p^{\mathcal{A}}.$$

Hence again, (9) holds true. □

## Lemma NUPRED

Let  $SP = (\Sigma, P, AX)$  be a co-Horn specification and  $\mathcal{A}$  be a  $\Sigma'$ -algebra with carrier  $A$  that satisfies  $AX$ .

Every  $\Sigma$ -homomorphism  $h : \mathcal{A}|_\Sigma \rightarrow \mathcal{C}$  is a  $\Sigma'$ -homomorphism from  $\mathcal{A}$  to  $gfp(\Phi)$ .

In particular, if  $\mathcal{C}$  is final in  $Alg_\Sigma$ , then  $gfp(\Phi)$  is final in  $Alg_{\Sigma', AX}$ .

*Proof.* It remains to show that for all  $p : e \in P$ ,

$$h(p^{\mathcal{A}}) \subseteq p^{gfp(\Phi)},$$

or, equivalently, by Lemma COCONSTEP and Kleene's Fixpoint Theorem (2), for all  $i \in \mathbb{N}$ ,

$$h(p^{\mathcal{A}}) \subseteq p^{\Phi^i(\top)}. \tag{10}$$

Let  $e = s_1 \times \cdots \times s_n$ .

*Case 1:*  $i = 0$ . Since  $p^\top = C_e$ , (10) holds true trivially.

*Case 2:* Let  $i > 0$  and  $c = (c_1, \dots, c_n) \in C_e \setminus p^{\Phi^i(\top)}$ .

Then there are  $t = (t_1, \dots, t_n) \in T_\Sigma(X)^n$ ,  $ax = (p(t) \Rightarrow \varphi) \in AX$  and

$$g \in C^V \setminus \varphi^{\Phi^{i-1}(\top)} \quad (11)$$

such that  $c = g^*(t)$ . Let  $X = \{x_1, \dots, x_n\}$  be a set of pairwise different variables disjoint from  $\text{var}(ax)$ . Let  $\{z_1, \dots, z_m\} = \text{var}(ax)$  and  $\psi = (\forall z_1 \dots \forall z_m (\varphi \vee \bigvee_{k=1}^n x_k \neq t_k))$ .

Obviously,  $ax$  is equivalent to the  $\Sigma'$ -formula  $ax' = (p(x_1, \dots, x_n) \Rightarrow \psi)$ .

W.l.o.g.  $g(x_k) = g^*(t_k)$  for all  $1 \leq k \leq n$ . It remains to show  $c \notin h(p^A)$ .

Hence assume that there is  $a = (a_1, \dots, a_n) \in p^A$  with  $c = h(a)$ . Let  $f \in A^V$  be such that  $f(x_k) = a_k$  for all  $1 \leq k \leq n$ . Then for all  $1 \leq k \leq n$ ,

$$h(f(x_k)) = h(a_k) = c_k = g^*(t_k) = g(x_k). \quad (12)$$

$f(x_1, \dots, x_n) = a \in p^A$  implies  $f \in p(x_1, \dots, x_n)^A$  and thus  $f \in \psi^A$  because  $A \models ax$  implies  $A \models ax'$ . By induction hypothesis,  $h$  is a  $\Sigma'$ -homomorphism from  $A$  to  $\Phi^{i-1}(\top)$ . Hence by Lemma **NEGFREE**,  $h \circ f \in \psi^{\Phi^{i-1}(\top)}$  and thus

$$h \circ f \in \psi^{\Phi^{i-1}(\top)}. \quad (13)$$

Since all variables of  $\text{var}(\psi) \setminus X$  are universally quantified, (12) and (13) imply

$$g \in (\varphi \vee \bigvee_{k=1}^n x_k \neq t_k)^{\Phi^{i-1}(\top)}$$

and thus  $g \in \varphi^{\Phi^{i-1}(\top)}$  because  $g(x_k) = g^*(t_k)$  for all  $1 \leq k \leq n$ .  $g \in \varphi^{\Phi^{i-1}(\top)}$  contradicts (11). Hence  $c \notin h(p^A)$ .  $\square$

## Theorem COMPLAX

Let  $SP = (\Sigma, P, AX)$  be a Horn or co-Horn specification and  $\Sigma' = \Sigma \cup P$ . Suppose that for all predicates  $p$  of  $\Sigma$ ,  $\bar{p}^C = C_e \setminus p^C$ .

Let  $coP = \{\bar{p} : e \rightarrow 2 \mid p : e \in P\}$ ,  $coSP = (\Sigma, coP, coAX)$ ,  $co\Sigma' = \Sigma \cup coP$  and

$$coAX = \begin{cases} \{\bar{p}(t) \Rightarrow \bar{\varphi} \mid p(t) \Leftarrow \varphi \in AX\} & \text{if } SP \text{ is a Horn specification,} \\ \{\bar{p}(t) \Leftarrow \bar{\varphi} \mid p(t) \Rightarrow \varphi \in AX\} & \text{if } SP \text{ is a co-Horn specification} \end{cases}$$

where  $\bar{\varphi}$  is defined inductively as follows: For all  $\Sigma'$ -atoms  $p(t)$  and  $\Sigma'$ -formulas  $\varphi, \psi$ ,  $\overline{p(t)} = \bar{p}(t)$ ,  $\overline{\varphi \wedge \psi} = \bar{\varphi} \vee \bar{\psi}$  and  $\overline{\varphi \vee \psi} = \bar{\varphi} \wedge \bar{\psi}$ .

Let  $\Phi = \Phi_{SP,C}$  and  $\Psi = \Phi_{coSP,C}$ .

(i) Let  $SP$  be a Horn specification. Then  $coSP$  is a co-Horn specification and for all  $p : e \rightarrow 2 \in P$ ,

$$\bar{p}^{gfp(\Psi)} = C_e \setminus p^{lfp(\Phi)}.$$

(ii) Let  $SP$  be a co-Horn specification. Then  $coSP$  is a Horn specification and for all  $p : e \rightarrow 2 \in P$ ,

$$\bar{p}^{lfp(\Psi)} = C_e \setminus p^{gfp(\Phi)}.$$

*Proof of (i).* At first, we show that for all negation- and quantifier-free  $\Sigma'$ -formulas  $\varphi$ ,  $\mathcal{A} \in Alg_{\Sigma',C}$  and  $\mathcal{B} \in Alg_{co\Sigma',C}$  such that for all  $p : e \rightarrow 2 \in P$ ,  $\bar{p}^{\mathcal{B}} = C_e \setminus p^{\mathcal{A}}$ ,

$$\overline{\varphi}^{\mathcal{B}} = C^V \setminus \varphi^{\mathcal{A}}. \quad (14)$$

We show (14) by induction on the size of  $\varphi$ . For all  $\Sigma'$ -atoms  $p(t)$ ,

$$\begin{aligned} \overline{p(t)}^{\mathcal{B}} &= \{g \in C^V \mid g^*(t) \in \bar{p}^{\mathcal{B}}\} = \{g \in C^V \mid g^*(t) \in C_e \setminus p^{\mathcal{A}}\} \\ &= C^V \setminus \{g \in C^V \mid g^*(t) \in p^{\mathcal{A}}\} = C^V \setminus p(t)^{\mathcal{A}}. \end{aligned}$$

For all negation- and quantifier-free  $\Sigma'$ -formulas  $\varphi, \psi$ ,

$$\begin{aligned} \overline{\varphi \wedge \psi}^{\mathcal{B}} &= (\overline{\varphi} \vee \overline{\psi})^{\mathcal{B}} = \overline{\varphi}^{\mathcal{B}} \cup \overline{\psi}^{\mathcal{B}} \stackrel{ind. \ hyp.}{=} (C^V \setminus \varphi^{\mathcal{A}}) \cup (C^V \setminus \psi^{\mathcal{A}}) \\ &= C^V \setminus (\varphi^{\mathcal{A}} \cap \psi^{\mathcal{A}}) = C^V \setminus (\varphi \wedge \psi)^{\mathcal{A}}. \end{aligned}$$

Now suppose that for all  $p : e \rightarrow 2 \in P$  and  $i \in \mathbb{N}$ ,

$$\bar{p}^{\Psi^i(\top)} = C_e \setminus p^{\Phi^i(\perp)}. \quad (15)$$

By Theorem CONSTEP,  $\Phi$  is  $\omega$ -continuous and  $\Psi$  is  $\omega$ -cocontinuous. Hence by Kleene's Fixpoint Theorem,

$$\bar{p}^{gfp(\Psi)} = \bigcap_{i<\omega} \bar{p}^{\Psi^i(\top)} \stackrel{(15)}{=} \bigcap_{i<\omega} (C_e \setminus p^{\Phi^i(\perp)}) = C_e \setminus \bigcup_{i<\omega} p^{\Phi^i(\perp)} = C_e \setminus p^{lfp(\Phi)}.$$

It remains to show (15). Let  $i = 0$ . Then

$$\bar{p}^{\Psi^i(\top)} = \bar{p}^\top = C_e = C_e \setminus \emptyset = C_e \setminus p^\perp = C_e \setminus p^{\Phi^i(\perp)}.$$

Let  $i > 0$ . Then by induction hypothesis, for all  $p : e \rightarrow 2 \in P$ ,  $\bar{p}^{\Psi^{i-1}(\top)} = C_e \setminus p^{\Phi^{i-1}(\perp)}$ . Hence by (14), for all negation- and quantifier-free  $\Sigma'$ -formulas  $\varphi$ ,

$$\bar{\varphi}^{\Psi^{i-1}(\top)} = C^V \setminus \varphi^{\Phi^{i-1}(\perp)}, \quad (16)$$

and thus

$$\begin{aligned} \bar{p}^{\Psi^i(\top)} &= C_e \setminus \{g^*(t) \mid \bar{p}t \Rightarrow \bar{\varphi} \in coAX, g \in C^V \setminus \bar{\varphi}^{\Psi^{i-1}(\top)}\} \\ &= C_e \setminus \{g^*(t) \mid pt \Leftarrow \varphi \in AX, g \in C^V \setminus \bar{\varphi}^{\Psi^{i-1}(\top)}\} \\ &\stackrel{(16)}{=} C_e \setminus \{g^*(t) \mid pt \Leftarrow \varphi \in AX, g \in C^V \setminus (C^V \setminus \varphi^{\Phi^{i-1}(\perp)})\} \\ &= C_e \setminus \{g^*(t) \mid pt \Leftarrow \varphi \in AX, g \in \varphi^{\Phi^{i-1}(\perp)}\} = C_e \setminus p^{\Phi^i(\perp)}. \end{aligned}$$

(ii) can be shown analogously. □

## Deduction in sequent logic

- **Top-down derivations** transform  $\Sigma$ -formulas of type 2 into *true* or other formulas that represent solutions:

*prove*  $\varphi$ :  $\varphi \vdash \text{true}$

*solve*  $\varphi$ :  $\varphi \vdash$  solved formula (see below)

*refute*  $\varphi$ :  $\neg\varphi \vdash \text{true}$

*verify*  $p$ :  $p(x) \Rightarrow \varphi \vdash \text{true}$

*evaluate*  $p$ :  $p(x) \Leftarrow \varphi \vdash \text{true}$

*evaluate*  $t$ :  $t = x \vdash x = u$

*reduce*  $t$ :  $t \rightarrow x \vdash \bigvee_{i=1}^n x = u_i$

A derivation

$$\varphi_1 \vdash \varphi_2 \vdash \dots \vdash \varphi_n$$

is sound with respect to the fixpoint semantics defined above, i.e., yields a sequence of reverse implications:

$$\varphi_1 \Leftarrow \varphi_2 \Leftarrow \dots \Leftarrow \varphi_n$$

The above goals are achieved if  $\varphi_1$  and  $\varphi_n$ , respectively, look as follows:

*prove*  $\varphi_1$ :  $\varphi_n = \text{True}$

*refute*  $\varphi_1$ :  $\varphi_n = \text{False}$

*solve*  $\varphi_1$ :  $\varphi_n$  is a **solved formula**, i.e.,

$$\varphi_n = \bigwedge_{i=1}^k \exists Z_i : x_i = u_i \wedge \bigwedge_{i=k+1}^r \forall Z_i : x_i \neq u_i$$

where  $x_1, \dots, x_r$  are different variables,

$u_1, \dots, u_r$  are irreducible “normal forms”,

and the relation  $\{(i, j) \mid u_i \text{ contains } x_j\}^+$  is acyclic.

*evaluate t*:  $\varphi_1 = (t = x)$ ,  $\varphi_n = (x = u)$

*reduce t*:  $\varphi_1 = (t \rightarrow x)$ ,  $\varphi_n = (x = u_1) \vee \dots \vee (x = u_k)$

Other derivations performed by Expander2 are sequences of (sets of)  $\Sigma$ -formulas of an arbitrary type:

$$t_1 \vdash t_{21} <+> \dots <+> t_{2k_2} \vdash \dots \vdash t_{n1} <+> \dots <+> t_{nk_n}$$

$\langle+\rangle$  is a built-in associative, commutative and idempotent operator that combines several reducts of the same redex. Its zero element () denotes *undefined*.

## Rules at three levels of automation/interaction

- *bottom*: **Simplifications** are equivalence transformations that partially evaluate terms and formulas.
- *medium*: **(Co)Resolution**, **narrowing** and **rewriting**, i.e., narrowing without proper redex instantiation, apply **axioms** to **goals** (formulas or terms), interactively or automatically, stepwise or iteratively.
- *top*: **Induction** and **coinduction** and other proper **expansion rules** are mostly used interactively and stepwise (see chapter 11). They apply **goals** (hypotheses) to **axioms** and thus prove the former by solving the latter.

## Rule applicability

Let  $\mathcal{A}$  be a  $\Sigma'$ -algebra and  $\varphi, C(\varphi), \psi$  be  $\Sigma'$ -formulas such that  $\varphi$  is a subformula of  $C(\varphi)$ .

- $\frac{\varphi}{\psi} \Downarrow$  denotes a **simplification rule for  $\mathcal{A}$** , i.e.,  $\mathcal{A}$  satisfies  $\psi \Leftrightarrow \varphi$ .

Applied in any context  $C[\varphi]$ , it leads to a further simplification rule:

$$\frac{C[\varphi]}{C[\psi]} \Downarrow$$

- $\frac{\varphi}{\psi} \uparrow$  denotes an **expansion rule for  $\mathcal{A}$** , i.e.,  $\mathcal{A}$  satisfies  $\psi \Rightarrow \varphi$ .

Applied in a context  $C[\varphi]$  where  $\varphi$  has positive polarity, it leads to a further expansion rule:

$$\text{polarity}(\text{position}(\varphi), C[\varphi]) = + \implies \frac{C[\varphi]}{C[\psi]} \uparrow$$

- $\frac{\varphi}{\psi} \Downarrow$  denotes a **contraction rule for  $\mathcal{A}$** , i.e.,  $\mathcal{A}$  satisfies  $\varphi \Rightarrow \psi$ .

Applied in a context  $C[\varphi]$  where  $\varphi$  has negative polarity, it leads to an expansion rule:

$$\text{polarity}(\text{position}(\varphi), C[\varphi]) = - \implies \frac{C[\varphi]}{C[\psi]} \uparrow$$

## Resolution and narrowing (see also [118, 127, 120])

The (co-)Horn clauses used by the following rules may have **guards**  $\gamma \in Fo_\Sigma(V)$ , which are those parts of the respective premises that must be solvable by the unifiers that trigger the rule applications.

In **Expander2**, (co)resolution steps are performed by pushing the *narrow* button. Unification of axioms with the actual goal is restricted to matching if the *match/unify* button left of the *narrow* button is set to *match*. The intervening *all/random* button admits to switch between the application of all applicable axioms in parallel (see the respective rule succedents) and the random selection of a single applicable rule.

- ❖ **Simplification rules** [120, 127] execute equivalence transformations of formulas and terms. Moreover, the simplifier of Expander2 partially evaluates terms w.r.t. built-in data types.
- ❖ **Narrowing rules** (including resolution and coresolution) apply axioms to formulas.
- ❖ **Rewriting rules**, i.e., narrowing rules without proper redex instantiation, apply axioms to terms.
- ❖ **Induction, coinduction** and other expansion or contraction rules (see above) are applied to formulas, always locally and stepwise.

- Resolution upon a least predicate  $p \in P$

Let  $\gamma_1 \Rightarrow (p(t_1) \Leftarrow \varphi_1), \dots, \gamma_n \Rightarrow (p(t_n) \Leftarrow \varphi_n)$  be all Horn clauses for  $p$  in  $AX$ ,

(\*)  $\vec{x}$  be a list of the variables of  $t$ ,

for all  $1 \leq i \leq k$ ,  $t\sigma_i = t_i\sigma_i$ ,  $\mathcal{C} \models \gamma_i\sigma_i$  and  $Z_i = var(t_i, \varphi_i)$ ,

for all  $k < i \leq n$ ,  $t$  be not unifiable with  $t_i$ .

$$\frac{p(t)}{\bigvee_{i=1}^k \exists Z_i : (\varphi_i\sigma_i \wedge \vec{x} = \vec{x}\sigma_i)} \updownarrow$$

If only a single axiom for  $p$  is applied, then the corresponding rule is only an expansion rule.

- Narrowing upon a function  $f \in F$

Let  $\gamma_1 \Rightarrow (f(t_1) = u_1 \Leftarrow \varphi_1), \dots, \gamma_n \Rightarrow (f(t_n) = u_n \Leftarrow \varphi_n)$  be all Horn clauses for  $f$  in  $AX$ ,  $at(x)$  be a  $\Sigma'$ -atom,

(\*\*)  $\vec{x}$  be a list of the variables of  $t$ ,

for all  $1 \leq i \leq k$ ,  $t_i\sigma_i = t\sigma_i$ ,  $\mathcal{C} \models \gamma_i\sigma_i$  and  $Z_i = var(t_i, u_i, \varphi_i)$ ,

for all  $k < i \leq l$ ,  $\sigma_i$  be a partial unifier of  $t$  and  $t_i$ ,

i.e.,  $t'_i \leq t_i$  and  $t'_i\sigma_i = t\sigma_i$  for some  $t'_i \in T_\Sigma(V) \setminus V$ ,

for all  $l < i \leq n$ ,  $t$  be not partially unifiable with  $t_i$ .

$$\frac{at(f(t))}{\bigvee_{i=1}^k \exists Z_i : (at(u_i)\sigma_i \wedge \varphi_i\sigma_i \wedge \vec{x} = \vec{x}\sigma_i) \vee \bigvee_{i=k+1}^l (at(f(t))\sigma_i \wedge \vec{x} = \vec{x}\sigma_i)} \Updownarrow$$

Again, if only a single axiom for  $f$  is applied, then the corresponding rule is only an expansion rule.

- **Narrowing upon a transition relation  $\rightarrow \in F$**

Let  $\gamma_1 \Rightarrow (t_1 \rightarrow u_1 \Leftarrow \varphi_1), \dots, \gamma_n \Rightarrow (t_n \rightarrow u_n \Leftarrow \varphi_n)$  be all Horn clauses for  $\rightarrow$  in  $AX$ ,  $\sigma_i$  be a unifier modulo associativity and commutativity of  $\wedge$  and (\*\*\*) hold true.

$$\frac{t \wedge v \rightarrow t'}{\bigvee_{i=1}^k \exists Z_i : ((u_i \wedge v)\sigma_i = t'\sigma_i \wedge \varphi_i\sigma_i \wedge \vec{x} = \vec{x}\sigma_i) \vee \bigvee_{i=k+1}^l ((t \wedge v)\sigma_i \rightarrow t'\sigma_i \wedge \vec{x} = \vec{x}\sigma_i)} \Updownarrow$$

Again, if only a single axiom for  $\rightarrow$  is applied, then the corresponding rule is only an expansion rule.

As pointed out in [14, 110, 111], partial unification is needed for ensuring the completeness of narrowing if the redex  $f(t)$  is selected according to outermost (“lazy”) strategies, which—as in the case of rewriting—are the only ones that guarantee termination and optimality.

- **Rewriting upon a function  $f \in F$**

Let  $f(t_1) = u_1 \Leftarrow \varphi_1, \dots, f(t_n) = u_n \Leftarrow \varphi_n$  be all Horn clauses for  $f$  in  $AX$   
 $(***)$  for all  $1 \leq i \leq k$ ,  $t = t_i\sigma_i$  and  $\mathcal{C} \models \gamma_i\varphi_i$ ,  
for all  $k < i \leq n$ ,  $t$  does not match  $t_i$ .

$$\frac{f(t)}{u_1\sigma_1 \text{---} \cdots \text{---} u_k\sigma_k}$$

- **Rewriting upon a transition relation  $\rightarrow \in F$**

Let  $t_1 \rightarrow u_1 \Leftarrow \varphi_1, \dots, t_n \rightarrow u_n \Leftarrow \varphi_n$  be all Horn clauses for  $\rightarrow$  in  $AX$  and  $(***)$  hold true.

$$\frac{t}{u_1\sigma_1 \text{---} \cdots \text{---} \cdots \text{---} \cdots \text{---} u_k\sigma_k}$$

- **Coresolution upon a greatest predicate  $p \in P$**

Let  $AX_p = \{\gamma_1 \Rightarrow (p(t_1) \Rightarrow \varphi_1), \dots, \gamma_n \Rightarrow (p(t_n) \Rightarrow \varphi_n)\}$  be all co-Horn clauses for  $p$  in  $AX$  and  $(*)$  hold true.

$$\frac{p(t)}{\bigwedge_{i=1}^k \forall Z_i : (\varphi_i \sigma_i \vee \vec{x} \neq \vec{x} \sigma_i)} \quad \Updownarrow$$

If only a single axiom for  $p$  is applied, then the corresponding rule is only a contraction rule.

- **Elimination of irreducible atoms and terms**

Let  $p$  be a least and  $q$  be a greatest predicate of  $P$ ,  $f \in F$  and  $\rightarrow$  be a binary predicate of  $F$ ,  $at$  be a  $\Sigma'$ -atom,  $p(t)$ ,  $q(t)$ ,  $f(t)$  and  $t \rightarrow t'$  be irreducible atoms resp. terms, i.e., none of the above rules is applicable.

$$\frac{p(t)}{False} \quad \frac{q(t)}{True} \quad \frac{at(f(t))}{at()} \quad \frac{t \rightarrow t'}{() \rightarrow t'} \quad \Updownarrow$$

The elimination rules are correct only if  $p$ ,  $q$ ,  $f$  and  $\rightarrow$  are axiomatized completely.

## Fixpoint induction and coinduction

Let  $\Sigma = (S, \mathcal{I}, F)$  and  $\Sigma' = (S, \mathcal{I}, F \cup P)$  be signatures and  $\mathcal{C}$  be a  $\Sigma$ -algebra such that  $P$  consists of predicates.

### Incremental induction upon a least predicate $p$

Let  $SP = (\Sigma, P, AX)$  be a Horn specification,  $p \in P$  and  $x \in V^+$ .

Fixpoint induction allows us to prove that  $\mathcal{A} = lfp(\Phi_{SP,C})$  satisfies a co-Horn clause  $p(x) \Rightarrow \varphi$  (see chapter 10).

A proof of  $p(x) \Rightarrow \varphi$  by fixpoint induction is a sequence  $(\psi_1, \dots, \psi_n)$  of  $\Sigma$ -formulas such that the following conditions hold true:

- $\psi_2$  is the result of applying the following rule to  $\psi_1$ :

$$(1) \quad \frac{p(x) \Rightarrow \varphi}{\bigwedge_{p(t) \leftarrow \delta \in AX} (\delta[q/p] \Rightarrow \varphi[t/x])} \uparrow \quad p \notin \varphi$$

After applying (1), the predicate  $q$  and the co-Horn clause  $q(x) \Rightarrow \varphi$  are added to  $SP$ .

- For all  $1 < i < n$ ,  $\psi_{i+1}$  is the result of applying to  $\psi_i$  an expansion rule for  $\mathcal{B}$  (see chapter 10) or the following rule:

$$(2) \quad \frac{q(x) \Rightarrow \varphi'}{\bigwedge_{p(t) \Leftarrow \delta \in AX} (\delta[q/p] \Rightarrow \varphi'[t/x])} \quad p, q \notin \varphi'$$

After applying (2), the co-Horn clause  $q(x) \Rightarrow \varphi'$  is added to  $SP$ .

- $\psi_n = True$ .

(1) is an expansion rule for  $\mathcal{A}$ : If the succedent of (1) holds true in  $\mathcal{A}$ , then  $\mathcal{A}$  satisfies the axioms for  $p$  if  $p$  were replaced by  $\varphi$ . Since  $\mathcal{A}$  interprets  $p$  as the *least* relation satisfying the axioms for  $p$ , we conclude that the antecedent of (1) holds true in  $\mathcal{A}$ .

Let  $\mathcal{B} \in Alg_{\Sigma', \mathcal{C}}$  such that  $p(x)^{\mathcal{B}} = \varphi^{\mathcal{C}}$ .

By Lemma IND,  $\mathcal{B}$  satisfies the succedent of (1) iff  $\mathcal{B}$  is  $\Phi_{SP, \mathcal{C}}$ -closed. Hence (1) can be rewritten as follows:

$$(1) \quad \frac{\mathcal{A} \leq \mathcal{B}}{\Phi_{SP, \mathcal{C}}(\mathcal{B}) \leq \mathcal{B}} \uparrow$$

Here the implication  $\uparrow$  follows from the definition of  $\Phi_{SP, \mathcal{C}}$  as the least  $\Phi_{SP, \mathcal{C}}$ -closed algebra of  $Alg_{SP, \mathcal{C}}$  (see chapter 10).

Proof sketch of the correctness of  $(\psi_1, \dots, \psi_n)$

Suppose that the derivation  $(\psi_1, \dots, \psi_n)$  contains  $k$  applications of (2). Then it reads schematically as follows:

$$\begin{array}{c}
 p(x) \Rightarrow \varphi \\
 \vdash^{(1)} \quad \bigwedge_{p(t) \Leftarrow \delta \in AX} (\delta[q/p] \Rightarrow \varphi[t/x]) \quad (*) \\
 \text{expansion rules} \\
 \vdash \quad \dots \ q(x) \Rightarrow \varphi_1 \ \dots \\
 \vdash^{(2)} \quad \dots \ \bigwedge_{p(t) \Leftarrow \delta \in AX} (\delta[q/p] \Rightarrow \varphi_1[t/x]) \ \dots \\
 \vdash \quad \dots \\
 \text{expansion rules} \\
 \vdash \quad \dots \ q(x) \Rightarrow \varphi_k \ \dots \\
 \vdash^{(2)} \quad \dots \ \bigwedge_{p(t) \Leftarrow \delta \in AX} (\delta[q/p] \Rightarrow \varphi_k[t/x]) \ \dots \\
 \text{expansion rules} \\
 \vdash \quad \text{True}
 \end{array}$$

Since  $q \notin \varphi \wedge \varphi_1 \wedge \dots \wedge \varphi_k$ ,  $q(x)$  is equivalent to  $\varphi$  before the first application of (2), while—due to the stepwise addition of axioms for  $q$  (see above)—for all  $1 \leq i \leq k$ ,  $q(x)$  is equivalent to  $\varphi \wedge \varphi_1 \wedge \dots \wedge \varphi_i$  after the  $i$ -th application of (2).

Since  $q$  occurs only in the premise of derived implications, the subderivation starting with  $(*)$  remains correct if, from the beginning,  $q(x)$  is considered to be equivalent to  $\varphi \wedge \varphi_1 \wedge \cdots \wedge \varphi_k$ . Then for all  $1 \leq i \leq k$ ,  $q(x) \Rightarrow \varphi_i$  holds true, and thus the subderivation starting with  $(*)$  yields the validity of

$$\bigwedge_{p(t) \Leftarrow \delta \in AX} (\delta[q/p] \Rightarrow \varphi[t/x]) \quad (3)$$

and

$$\bigwedge_{p(t) \Leftarrow \delta \in AX} \bigwedge_{i=1}^k (\delta[q/p] \Rightarrow \varphi_i[t/x]). \quad (4)$$

$(3) \wedge (4)$  is equivalent to

$$\bigwedge_{p(t) \Leftarrow \delta \in AX} (\delta[q/p] \Rightarrow (\varphi \wedge \varphi_1 \wedge \cdots \wedge \varphi_k)[t/x]) \text{ and thus to } \bigwedge_{p(t) \Leftarrow \delta \in AX} (\delta[q/p] \Rightarrow q[t/x]).$$

Hence  $q$  (instead of  $p$ ) satisfies  $AX$  in  $\mathcal{A}$  and thus by the correctness of (1),

$$p(x) \Rightarrow q(x) \quad (5)$$

and, in particular, the original goal  $p(x) \Rightarrow \varphi$  hold true in  $\mathcal{A}$ .

$q(x)$  can be regarded as a **generalization** of  $\varphi$ . By (5),  $q(x)$  lies *somewhere* between  $p(x)$  and  $\varphi$ , the least (!) model of  $AX$ :

$$p(x) \Rightarrow q(x) \Rightarrow \varphi.$$

Hence the validity of an inductive conjecture like  $p(x) \Rightarrow \varphi$  is not semi-decidable, let alone decidable.

If  $p$  were a *greatest* predicate, then proving conjectures of the form  $p(x) \Rightarrow \varphi$  amounts to coresolving them upon  $p$  (see [Duality of \(co\)resolution and \(co\)induction](#)).

For verifying a recursive function  $f$ , i.e., showing properties of their input-output relation, we transform the Horn clauses for  $f : e \rightarrow e'$  into Horn clauses for  $p_f : e \times e' \rightarrow 2$ —representing  $graph(f)$  (see [Preliminaries](#))—by repeatedly applying the following transformation rules:

$$\frac{f(t) = u \Leftarrow \varphi}{p_f(t, u) \Leftarrow \varphi} \Updownarrow$$

$$\frac{\psi[f(v)/x] \Leftarrow \varphi}{\psi \Leftarrow p_f(v, x) \wedge \varphi} \Updownarrow \quad \frac{\psi \Leftarrow \varphi[f(v)/x]}{\psi \Leftarrow \varphi \wedge p_f(v, x)} \Updownarrow$$

Let  $rel(AX)$  be a set of Horn clauses for  $p_f$  each of which is obtained from applying the above rules to  $AX$  and does not contain  $f$ . Hence (1) and (2) entail the following rules:

### Incremental induction upon a function $f$

$$(1) \quad \frac{f(x) = y \Rightarrow \varphi}{\bigwedge_{p_f(t,u) \Leftarrow \delta \in rel(AX)} (\delta[q/p_f] \Rightarrow \varphi[t/x, u/y])} \uparrow \quad f, p_f \notin \varphi$$

After applying (1), the predicate  $q$  and the co-Horn clause  $q(x, y) \Rightarrow \varphi$  are added to  $SP$ .

$$(2) \quad \frac{q(x, y) \Rightarrow \varphi'}{\bigwedge_{p_f(t,u) \Leftarrow \delta \in rel(AX)} (\delta[q/p] \Rightarrow \varphi'[t/x, u/y])} \quad f, p_f, q \notin \varphi'$$

After applying (2), the co-Horn clause  $q(x, y) \Rightarrow \varphi'$  is added to  $SP$ .

**Expander2** applies (1) even to formulas that do not match the premise of (1), but can be turned into matching ones via the following **stretch rules**:

$$\frac{p(t) \Rightarrow \varphi}{p(x) \Rightarrow (x = t \Rightarrow \varphi)} \Updownarrow$$

$$\frac{f(t) = u \Rightarrow \varphi}{f(x) = y \Rightarrow (x = t \wedge y = u \Rightarrow \varphi)} \Updownarrow \quad \frac{f(t) = u \wedge \varphi}{f(x) = y \Rightarrow (x = t \Rightarrow y = u \wedge \varphi)} \Updownarrow$$

We leave it to the reader to adapt (1) and (2) to the case where several co-Horn clauses  $p_i(x) \Rightarrow \varphi_i$  or  $f_i(x) = y \Rightarrow \varphi_i$ ,  $1 \leq i \leq n$ , with least predicates  $p_i$  resp. functions  $f_i$  must be proved simultaneously because the Horn clauses for  $p_i$  resp.  $f_i$  provide a *mutually*-recursive definition.

Many examples of proofs by fixpoint induction can be found in [120, 127, 114].

## Incremental coinduction upon a greatest predicate $p$

Let  $SP = (\Sigma, P, AX)$  be a co-Horn specification,  $p \in P$  and  $x \in V^+$ .

Fixpoint coinduction allows us to prove that  $\mathcal{A} = gfp(\Phi_{SP,C})$  satisfies a Horn clause  $\varphi \Rightarrow p(x)$  (see chapter 10).

A proof of  $\varphi \Rightarrow p(x)$  by coinduction is a sequence  $(\psi_1, \dots, \psi_n)$  of  $\Sigma$ -formulas such that the following conditions hold true:

- $\psi_2$  is the result of applying to  $\psi_1$  the following rule:

$$(1) \quad \frac{\varphi \Rightarrow p(x)}{\Lambda_{p(t) \Rightarrow \delta \in AX}(\varphi[t/x] \Rightarrow \delta[q/p])} \quad p \notin \varphi$$

After applying (1), the predicate  $q$  and the Horn clause  $q(x) \Leftarrow \varphi$  are added to  $SP$ .

- For all  $1 < i < n$ ,  $\psi_{i+1}$  is the result of applying to  $\psi_i$  an expansion rule for  $\mathcal{B}$  (see chapter 10) or the following rule:

$$(2) \quad \frac{\varphi' \Rightarrow q(x)}{\Lambda_{p(t) \Rightarrow \delta \in AX}(\varphi'[t/x] \Rightarrow \delta[q/p])} \quad p, q \notin \varphi'$$

After applying (2), the Horn clause  $q(x) \Leftarrow \varphi'$  is added to  $SP$ .

- $\psi_n = True$ .

(1) is an expansion rule for  $\mathcal{A}$ : If the succedent of (1) holds true in  $\mathcal{A}$ , then  $\mathcal{A}$  satisfies the axioms for  $p$  if  $p$  were replaced by  $\varphi$ . Since  $\mathcal{A}$  interprets  $p$  as the *greatest* relation satisfying the axioms for  $p$ , we conclude that the antecedent of (1) holds true in  $\mathcal{A}$ .

Let  $\mathcal{B} \in Alg_{\Sigma', \mathcal{C}}$  such that  $p(x)^{\mathcal{B}} = \varphi^{\mathcal{C}}$ .

By Lemma COIND,  $\mathcal{A}$  satisfies the succedent of (1) iff  $\mathcal{B}$  is  $\Phi_{SP, \mathcal{C}}$ -dense. Hence (1) can be rewritten as follows:

$$(1) \quad \frac{\mathcal{B} \leq \mathcal{A}}{\mathcal{B} \leq \Phi_{SP, \mathcal{C}}(\mathcal{B})} \uparrow$$

Here the implication  $\uparrow$  follows from the definition of  $\Phi_{SP, \mathcal{C}}$  as the greatest  $\Phi_{SP, \mathcal{C}}$ -closed algebra of  $Alg_{SP, \mathcal{C}}$  (see chapter 10).

Proof sketch of the correctness of  $(\psi_1, \dots, \psi_n)$

Suppose that the derivation  $(\psi_1, \dots, \psi_n)$  contains  $k$  applications of (2). Then it schematically reads as follows:

$$\begin{array}{c} \varphi \Rightarrow p(x) \\ \vdash^{(1)} \bigwedge_{p(t) \Rightarrow \delta \in AX} (\varphi[t/x] \Rightarrow \delta[q/p]) \end{array} \quad (*)$$

*expansion rules*

$$\vdash \dots \varphi_1 \Rightarrow q(x) \dots$$

$(2)$

$$\vdash \dots \bigwedge_{p(t) \Rightarrow \delta \in AX} (\varphi_1[t/x] \Rightarrow \delta[q/p]) \dots$$

$\vdash$

$\dots$

*expansion rules*

$$\vdash \dots \varphi_k \Rightarrow q(x) \dots$$

$(2)$

$$\vdash \dots \bigwedge_{p(t) \Rightarrow \delta \in AX} (\varphi_k[t/x] \Rightarrow \delta[q/p]) \dots$$

*expansion rules*

$$\vdash True$$

Since  $q \notin \varphi \wedge \varphi_1 \wedge \dots \wedge \varphi_k$ ,  $q(x)$  is equivalent to  $\varphi$  before the first application of (2), while—due to the stepwise addition of axioms for  $q$  (see above)—for all  $1 \leq i \leq k$ ,  $q(x)$  is equivalent to  $\varphi \vee \varphi_1 \vee \dots \vee \varphi_i$  after the  $i$ -th application of (2).

Since  $q$  occurs only in the conclusion of derived implications, the subderivation starting with (\*) remains correct if, from the beginning,  $q(x)$  is considered to be equivalent to  $\varphi \vee \varphi_1 \vee \dots \vee \varphi_k$ . Then for all  $1 \leq i \leq k$ ,  $\varphi_i \Rightarrow q(x)$  holds true, and thus the subderivation starting with (\*) yields the validity of

$$\bigwedge_{p(t) \Rightarrow \delta \in AX} (\varphi[t/x] \Rightarrow \delta[q/p]) \tag{3}$$

and

$$\bigwedge_{p(t) \Rightarrow \delta \in AX} \bigwedge_{i=1}^k (\varphi_i[t/x] \Rightarrow \delta[q/p]). \quad (4)$$

(3)  $\wedge$  (4) is equivalent to

$$\bigwedge_{p(t) \Rightarrow \delta \in AX} ((\varphi \vee \varphi_1 \vee \cdots \vee \varphi_k)[t/x] \Rightarrow \delta[q/p]) \quad \text{and thus to} \quad \bigwedge_{p(t) \Rightarrow \delta \in AX} (q[t/x] \Rightarrow \delta[q/p]).$$

Hence by  $q$  (instead of  $p$ ) satisfies  $AX$  in  $\mathcal{A}$  and thus by the correctness of (1),

$$q(x) \Rightarrow p(x) \quad (5)$$

and, in particular, the original goal  $\varphi \Rightarrow p(x)$  hold true in  $\mathcal{A}$ .

$q(x)$  can be regarded as a **generalization** of  $\varphi$ . By (5),  $q(x)$  lies *somewhere* between  $\varphi$  and  $p(x)$ , the greatest (!) model of  $AX$ :

$$\varphi \Rightarrow q(x) \Rightarrow p(x).$$

Hence the validity of a coinductive conjecture like  $\varphi \Rightarrow p(x)$  is not semi-decidable, let alone decidable.

If  $p$  were a *least* predicate, then proving conjectures of the form  $\varphi \Rightarrow p(x)$  amounts to resolving them upon  $p$  (see Duality of (co)resolution and (co)induction).

Let  $p$  be a *binary* predicate and  $D\Sigma = (S, \mathcal{I}, D)$  be a—mostly, but not necessarily destructive—subsignature of  $\Sigma$  such that the set  $AX_p$  of co-Horn clauses for  $p$  consists of  $\Sigma''$ -bisimulation axioms, i.e.,  $\mathcal{B} \in Alg_{\Sigma', \mathcal{C}}$  satisfies  $AX_p$  iff  $p^{\mathcal{B}}$  is a  $\Sigma''$ -bisimulation on  $\mathcal{C}$ . Then the premise of (1) reads as

$$\varphi \Rightarrow p(x, y), \quad (6)$$

$p^A$  is the *greatest*  $D\Sigma$ -bisimulation on  $\mathcal{C}$ , while the above-sketched derivation proves that  $\mathcal{B} \in Alg_{\Sigma', \mathcal{C}}$  with  $p^{\mathcal{B}} = Q =_{def} (\varphi \vee \varphi_1 \vee \dots \vee \varphi_k)^{\mathcal{B}}$  also satisfies  $AX_p$ , i.e.,  $Q$  is also a  $D\Sigma$ -bisimulation on  $\mathcal{C}$ . Hence  $Q \subseteq p^A$ . If, in addition to the axioms for  $q$  that were added to  $AX$  after applications of (1) or (2), the Horn clauses

$$q(x, x), \quad q(x, y) \Leftarrow q(y, x), \quad q(x, y) \Leftarrow q(x, z) \wedge q(z, y) \quad (7)$$

were used in the proof of (6),  $q$  would actually denote the equivalence closure of  $Q$ , i.e.,

$$Q^{eq} \subseteq p^A \quad (8)$$

would actually be proved and not  $Q \subseteq p^A$ , which does not only follow from (8), but is even equivalent to (8):

Since  $Q \subseteq p^A$  implies  $Q^{eq} \subseteq (p^A)^{eq}$  and, by Theorem **BISIM** (2),  $p^A$  is an equivalence relation and thus equal to  $(p^A)^{eq}$ , (8) indeed follows from  $Q \subseteq p^A$ .

Let  $D\Sigma$  be destructive,  $\mathcal{C}$  be final in  $Alg_{D\Sigma}$ ,  $C\Sigma = (S, \mathcal{I}, C)$  be a constructive subsignature of  $\Sigma$  and for all  $s \in S$ , let  $d_s = \langle d \rangle_{d:s \rightarrow e \in D}$ . Moreover, suppose that

$$(d_s \circ c = f_c \circ d_e)_{s \in S}$$

is a biinductive definition of  $C$  on  $\mathcal{C}$  (see Theorem **RECFUN3**). If, in addition to the axioms for  $q$  that were added to  $AX$  after applications of (1) or (2), (8) and the Horn clauses

$$q(c(x), c(y)) \Leftarrow q(x, y), \quad c \in C, \tag{9}$$

were used in the proof of (6),  $q$  would actually denote the  $C\Sigma$ -congruence closure of  $Q$  (see **Bisimulation modulo constructors**), i.e.,

$$Q_C \subseteq p^A \tag{10}$$

would actually be proved and not  $Q \subseteq p^A$ , which does not only follow from (10), but is even equivalent to (10):

Since  $p^A$  is a  $D\Sigma$ -bisimulation on  $\mathcal{C}$ , Lemma **MOD** implies that  $p_C^A$  is also a  $\Sigma''$ -bisimulation. Since  $p^A$  is the *greatest* one,  $p^A \subseteq p_C^A \subseteq p^A$ , i.e.,  $p^A = p_C^A$ . Hence (10) indeed follows from  $Q \subseteq p^A$ .

If (8) is used in the proof, the latter is called a proof by **coinduction modulo  $C$** . For instance, the fact that the concatenation of regular languages distributes over summation, can be proved by coinduction modulo regular operators (see Example SEQPAR in chapter 15).

**Expander2** applies (1) even to formulas that do not match the premise of (1), but can be turned into matching ones via the following **stretch rule**:

$$\frac{\varphi \Rightarrow p(t)}{\varphi \wedge x = t \Rightarrow p(x)} \Updownarrow$$

We leave it to the reader to adapt (1) and (2) to the case where several Horn clauses  $\varphi_i \Rightarrow p_i$ ,  $1 \leq i \leq n$ , with greatest predicates  $p_i$  must be proved simultaneously because the co-Horn clauses for  $p_i$  provide a *mutually-recursive* definition.

Many examples of proofs by fixpoint coinduction can be found in chapter 15 and [120, 127].

*Coinductive logic programming* or *co-logic programming* [62, 154] has not much to do with coinduction. It is rather (co)resolution upon least or greatest predicates on models consisting of finite or infinite terms, respectively.

In contrast to the above (co)resolution rules, co-logic programming does not only resolve axioms upon (atoms of) the current goal  $\varphi$ , but also compares  $\varphi$  with all predecessors of  $\varphi$  in order to detect circularities in the derivation. We claim that most results obtained due to this—rather inefficient—inspection of the entire derivation would also be accomplished if the above (co)induction rules were used instead.

## Duality of (co)resolution and (co)induction

(Co)Resolution and narrowing upon functions apply axioms to conjectures.

The proof proceeds by transforming the modified conjectures.

(Co)Induction applies conjectures to axioms.

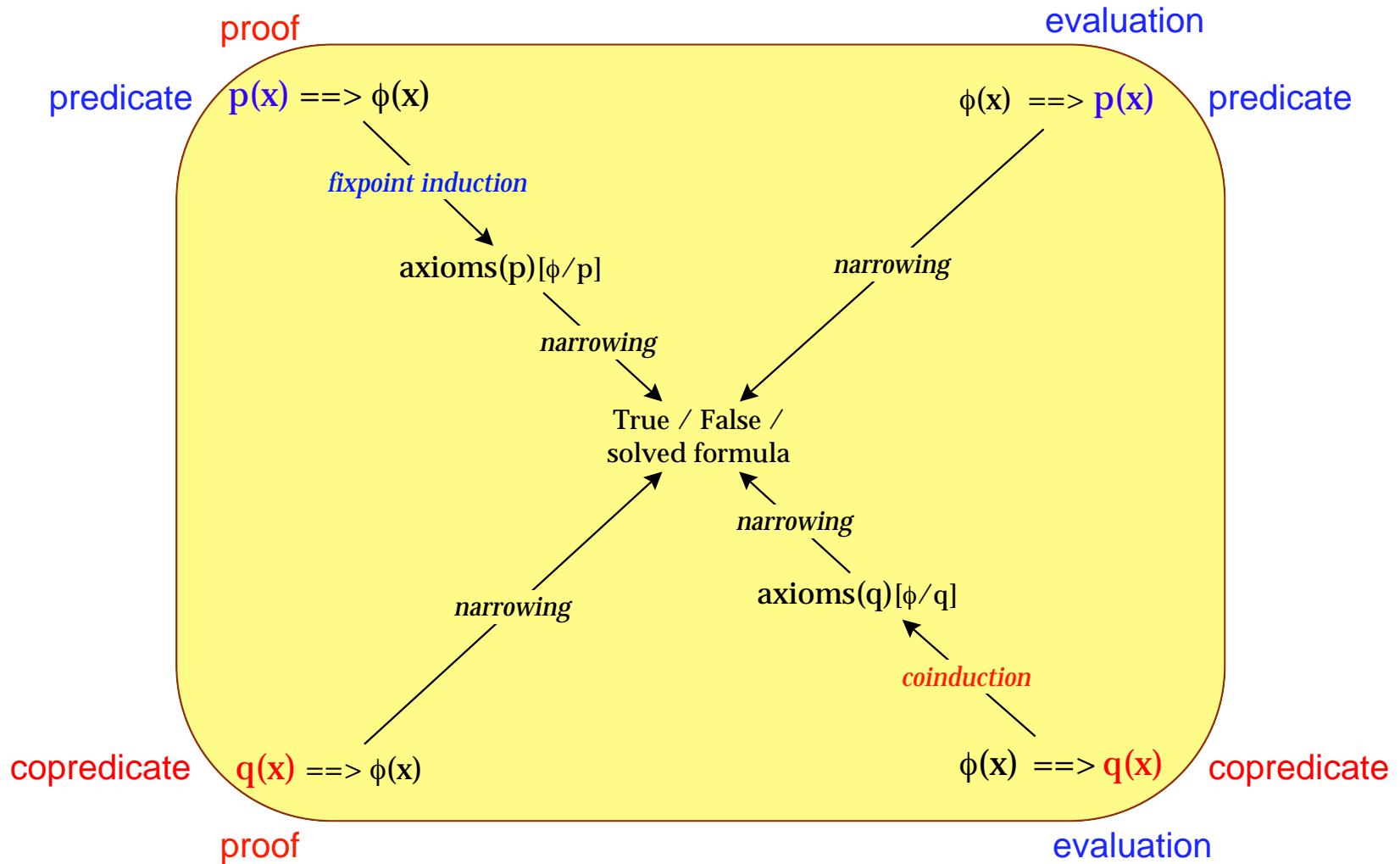
The proof proceeds by transforming the modified axioms.

Resolution upon a least predicate  $p$  is a rule for evaluating  $p$ .

Induction upon  $p$  is a rule for verifying  $p$ .

Coresolution upon a copredicate (greatest predicate)  $q$  is a rule for verifying  $q$ .

Coinduction upon  $q$  is a rule for evaluating  $q$ .



## *F*-algebras and *F*-coalgebras

Let  $\mathcal{K}$  be a category and  $F$  be an endofunctor on  $\mathcal{K}$ .

An  **$F$ -algebra** or  **$F$ -dynamics** [17] is a  $\mathcal{K}$ -morphism  $\alpha : F(A) \rightarrow A$ .

$\text{Alg}_F$  denotes the category of  $F$ -algebras and the following  $\mathcal{K}$ -morphisms:

An  **$\text{Alg}_F$ -morphism**  $h$  from an  $F$ -algebra  $\alpha : F(A) \rightarrow A$  to an  $F$ -algebra  $\beta : F(B) \rightarrow B$  is a  $\mathcal{K}$ -morphism  $h : A \rightarrow B$  such that  $h \circ \alpha = \beta \circ F(h)$ , i.e., the following diagram commutes:

$$\begin{array}{ccc} F(A) & \xrightarrow{\alpha} & A \\ F(h) \downarrow & = & \downarrow h \\ F(B) & \xrightarrow{\beta} & B \end{array}$$

An  **$F$ -coalgebra** or  **$F$ -codynamics** [17] is a  $\mathcal{K}$ -morphism  $\alpha : A \rightarrow F(A)$ .

$coAlg_F$  denotes the category of  $F$ -coalgebras and the following  $\mathcal{K}$ -morphisms:

A  **$coAlg_F$ -morphism**  $h$  from an  $F$ -coalgebra  $\alpha : A \rightarrow F(A)$  to an  $F$ -coalgebra  $\beta : B \rightarrow F(B)$  is a  $\mathcal{K}$ -morphism  $h : A \rightarrow B$  such that  $F(h) \circ \alpha = \beta \circ h$ , i.e., the following diagram commutes:

$$\begin{array}{ccc}
 A & \xrightarrow{\alpha} & F(A) \\
 h \downarrow & = & \downarrow F(h) \\
 B & \xrightarrow{\beta} & F(B)
 \end{array}$$

A  $\mathcal{K}$ -object  $A$  is a **fixpoint of  $F$**  if  $F(A) \cong A$ .

**Lambek's Lemma** ([92], Lemma 2.2; [23], Prop. 5.12; [15], section 2; [143], Thm. 9.1)

(1) Suppose that  $Alg_F$  has an initial object  $\alpha : F(A) \rightarrow A$ .

$\alpha$  is iso and thus  $A$  is a fixpoint of  $F$ .

(2) Suppose that  $coAlg_F$  has a final object  $\beta : A \rightarrow F(A)$ .

$\beta$  is iso and thus  $A$  is a fixpoint of  $F$ .

*Proof.* (1) Since  $\alpha$  is initial, there is an  $Alg_F$ -morphism  $h : A \rightarrow F(A)$  from  $\alpha$  to  $F(\alpha)$ .

Hence  $\alpha \circ h$  is an  $Alg_F$ -morphism from  $\alpha$  to  $\alpha$ :

$$\alpha \circ h \circ \alpha = \alpha \circ F(\alpha) \circ F(h) = \alpha \circ F(\alpha \circ h).$$

$id_A$  is also an  $Alg_F$ -morphism from  $\alpha$  to  $\alpha$ :

$$id_A \circ \alpha = \alpha = \alpha \circ id_{F(A)} = \alpha \circ F(id_A).$$

Hence (3)  $\alpha \circ h = id_A$  because  $\alpha$  is initial in  $Alg_F$ . Since  $h$  is an  $Alg_F$ -morphism,

$$h \circ \alpha = F(\alpha) \circ F(h) = F(\alpha \circ h) = F(id_A) = id_{F(A)}. \quad (4)$$

By (3) and (4),  $\alpha$  is an isomorphism.

(2) Analogously. □

Let  $\alpha : F(A) \rightarrow A$  be an  $F$ -algebra and  $ini_F : F(\mu F) \rightarrow \mu F$  be initial in  $Alg_F$ .

The unique  $Alg_F$ -morphism from  $ini_F$  to  $\alpha$  is called a **catamorphism** [100, 167], **reachability map** [17] or a function **defined by recursion** and denoted by  $fold^\alpha$  or  $(|\alpha|) : \mu F \rightarrow A$ .

Catamorphisms are also called functions **defined by recursion** because the equation that expresses that  $fold^\alpha$  is an  $Alg_F$ -morphism provides a recursive definition schema (see Theorem RECFUN1 and the section [Sample inductively defined functions](#)).

**Lemma PARA** Let  $\beta : F(A \times \mu F) \rightarrow A$  be a  $\mathcal{K}$ -morphism and  $\gamma$  be the  $F$ -algebra

$$\langle \beta, ini_F \circ F(\pi_2) \rangle : F(A \times \mu F) \rightarrow A \times \mu F.$$

There is a unique  $\mathcal{K}$ -morphism  $h : \mu F \rightarrow A$  such that (1) commutes:

$$\begin{array}{ccccc}
 F(\mu F) & \xrightarrow{\quad ini_F \quad} & \mu F & \xrightarrow{\quad \langle h, id \rangle \quad} & A \times \mu F \\
 \downarrow F(\langle h, id \rangle) & & \downarrow h & & \searrow \pi_1 \\
 F(A \times \mu F) & \xrightarrow{\quad \beta \quad} & A & &
 \end{array}
 \quad (1)$$

*Proof.* Suppose that (1) holds true. Then

$$\begin{aligned}
 \langle h, id \rangle \circ ini_F &\stackrel{(6) \text{ on } p. 16}{=} \langle h \circ ini_F, id \circ ini_F \rangle \\
 &\stackrel{id \text{ is } Alg_F\text{-morph.}}{=} \langle h \circ ini_F, ini_F \circ F(id) \rangle = \langle h \circ ini_F, ini_F \circ F(\pi_2 \circ \langle h, id \rangle) \rangle \\
 &= \langle h \circ ini_F, ini_F \circ F(\pi_2) \circ F(\langle h, id \rangle) \rangle \\
 &\stackrel{(1)}{=} \langle \beta \circ F(\langle h, id \rangle), ini_F \circ F(\pi_2) \circ F(\langle h, id \rangle) \rangle \\
 &\stackrel{(6) \text{ on } p. 16}{=} \langle \beta, ini_F \circ F(\pi_2) \rangle \circ F(\langle h, id \rangle) = \gamma \circ F(\langle h, id \rangle).
 \end{aligned}$$

Hence  $\langle h, id \rangle : \mu F \rightarrow A \times \mu F$  is an  $Alg_F$ -morphism from  $ini_F$  to  $\gamma$ . Given a further  $\mathcal{K}$ -morphism  $h : \mu F \rightarrow A$  such that (1) holds true with  $h'$  instead of  $h$ , the fact that  $\langle h, id \rangle$  is an  $Alg_F$ -morphism, can be shown analogously. Since there is only one  $Alg_F$ -morphism from  $ini_F$  to  $\gamma$ ,  $h = \pi_1 \circ \langle h, id \rangle = \langle h', id \rangle = h'$ . □

$h$  is called a **paramorphism** and denoted by  $\langle |\beta| \rangle$ .

Paramorphisms are the functions defined by **primitive recursion**. They match a particular recursion schema that can be reduced to (1). Such schema transformations often employ the step from given functions to their coextensions with respect to an adjunction (see chapter 17).

As to primitive recursion, the right-adjointness of products to diagonals provides the reduction: In terms of Theorem RECFUN4,  $h : \mu F \rightarrow A$  is a paramorphism iff

$$(h, id) : (\mu F, \mu F) \rightarrow (A, \mu F)$$

is  $(\Delta, \_ \times \mu F)$ -recursive, i.e., iff  $(h, id)^\# = \langle h, id \rangle$  is an  $Alg_F$ -morphism.

In the section Sample inductively defined functions, many recursion schemas and their reduction to (1) are exemplified. For instance, the equations given there for the factorial function yield a paramorphism.

Let  $\alpha : A \rightarrow F(A)$  be an  $F$ -coalgebra and  $fin_F : \nu F \rightarrow F(\nu F)$  be final in  $coAlg_F$ .

The unique  $coAlg_F$ -morphism from  $\alpha$  to  $\beta$  is called an **anamorphism** [100, 167] or **observability map** [17] and denoted by  $unfold^\alpha$  or  $|( \alpha )| : A \rightarrow \nu F$ .

Anamorphisms are also called functions **defined by corecursion** because the equation that expresses that  $unfold^\alpha$  is a  $coAlg_F$ -morphism provides a corecursive definition schema (see Theorem RECFUN2 and the section Sample coinductively defined functions).

**Lemma APO** Let  $\beta : A \rightarrow F(A + \nu F)$  be a  $\mathcal{K}$ -morphism and  $\gamma$  be the  $F$ -coalgebra

$$[\beta, F(\iota_2) \circ fin_F] : A + \nu F \rightarrow F(A + \nu F).$$

There is a unique  $\mathcal{K}$ -morphism  $h : A \rightarrow \nu F$  such that (2) commutes:

$$\begin{array}{ccc}
 A & \xrightarrow{\beta} & F(A + \nu F) \\
 \downarrow h & \swarrow \iota_1 & \downarrow F([h, id]) \\
 A + \nu F & \xrightarrow{[h, id]} & \nu F \xrightarrow{fin_F} F(\nu F)
 \end{array}
 \quad (2)$$

*Proof.* Analogously to the proof of Theorem PARA. □

$h$  is called an **apomorphism** and denoted by  $|\langle \beta \rangle|$ .

Apomorphisms are the functions defined by **primitive corecursion**. They match a particular corecursion schema that can be reduced to (2). Such schema transformations often employ the step from given functions to their extensions with respect to an adjunction (see chapter 17).

As to primitive corecursion, the left-adjointness of coproducts (sums) to diagonals provides the reduction: In terms of Theorem [RECFUN5](#),  $h$  is an apomorphism iff

$$(h, id) : (A, \nu F) \rightarrow (\nu F, \nu F)$$

is  $(\_ + \mu F, \Delta)$ -corecursive, i.e., iff  $(h, id)^* = [h, id]$  is an  $Alg_F$ -morphism.

In the sections [Sample coinductively defined functions](#) and [Sample biinductively defined functions](#), several corecursion schemas and their reduction to (2) are exemplified. For instance, the equations given there for a function that inserts elements into ordered streams yield an apomorphism.

Let  $ini_F : F(\mu F) \rightarrow \mu F$  be initial in  $Alg_F$  and  $fin_F : \nu F \rightarrow F(\nu F)$  be final in  $coAlg_F$  such that  $\mu F$  embedded in  $\nu F$ . Moreover, let  $Fin_F$  be the subcategory of  $coAlg_F$  that consists of all  $F$ -coalgebras  $\alpha : A \rightarrow F(A)$  such that  $unfold^\alpha : A \rightarrow \nu F$  factors through  $\mu F$ .

Let  $\alpha : F(A) \rightarrow A$  be an  $F$ -algebra and  $\beta : B \rightarrow F(B)$  be an  $F$ -coalgebra. A  $\mathcal{K}$ -morphism  $h : B \rightarrow A$  is a **hylo(morphism)** w.r.t.  $(\alpha, \beta)$  if

$$h = \alpha \circ F(h) \circ \beta \tag{3}$$

(see [73], section 3). If  $h$  is unique with (3), we write  $[\alpha, \beta]$  for  $h$  (see [100, 45]).

$\beta$  is **recursive** [9, 38] if for every  $F$ -algebra  $\alpha : F(A) \rightarrow A$  there is a unique hylo w.r.t.  $(\alpha, \beta)$ .

$\alpha$  is **corecursive** [8, 39] if for every  $F$ -coalgebra  $\beta : B \rightarrow F(B)$  there is a unique hylo w.r.t.  $(\alpha, \beta)$ .

Hence

- by Lambek's Lemma (1) and Lemma ISO (1), the inverse of an initial  $F$ -algebra  $ini_F : F(\mu F) \rightarrow \mu F$  is a recursive  $F$ -coalgebra with hylo

$$[|\alpha, ini_F^{-1}|] = (|\alpha|) : \mu F \rightarrow A; \quad (4)$$

- by Lambek's Lemma (2) and Lemma ISO (2), the inverse of a final  $F$ -coalgebra  $fin_F : \nu F \rightarrow F(\nu F)$  is a corecursive  $F$ -algebra with hylo

$$[|fin_F^{-1}, \beta|] = |(\beta)| : B \rightarrow \nu F. \quad (5)$$

**Lemma HYCO** (Hylo-Compose [52, 45])

Let  $\alpha : F(A) \rightarrow A$  and  $\beta' : F(B) \rightarrow B$  be  $F$ -algebras and  $\beta : B \rightarrow F(B)$ ,  $\gamma : C \rightarrow F(C)$  be  $F$ -coalgebras such that  $\beta \circ \beta' = id_{F(B)}$  and there are a hylo  $g : B \rightarrow A$  w.r.t.  $(\alpha, \beta)$  and a hylo  $h : C \rightarrow B$  w.r.t.  $(\beta', \gamma)$ .

(6)  $g \circ h : C \rightarrow A$  is a hylo w.r.t.  $(\alpha, \gamma)$ .

(7) Let  $ini_F : F(B) \rightarrow B$  be initial in  $Alg_F$ ,  $fin_F : B \rightarrow F(B)$  be final in  $coAlg_F$  and  $ini_F^{-1} = fin_F$  (or  $fin_F^{-1} = ini_F$ ).

Then  $(|\alpha|) \circ |(\gamma)|$  is a hylo w.r.t.  $(\alpha, \gamma)$ .

*Proof.* (6):

$$\begin{aligned} g \circ h &\stackrel{g \text{ hylo}}{=} \alpha \circ F(g) \circ \beta \circ h \stackrel{h \text{ hylo}}{=} \alpha \circ F(g) \circ \beta \circ \beta' \circ F(h) \circ \gamma \\ &\stackrel{\beta \circ \beta' = id}{=} \alpha \circ F(g) \circ F(h) \circ \gamma = \alpha \circ F(g \circ h) \circ \gamma. \end{aligned}$$

(7): By (4),  $(|\alpha|) = [|\alpha, ini_F^{-1}|] \stackrel{ini_F^{-1} = fin_F}{=} [|\alpha, fin_F|]$ . By (5),  $|(\gamma)| = [|fin_F^{-1}, \gamma|]$ . Hence  $g = (|\alpha|)$  and  $h = |(\gamma)|$  satisfy the assumptions of the lemma.

Therefore, (6) implies that  $(|\alpha|) \circ |(\gamma)|$  is a hylo w.r.t.  $(\alpha, \gamma)$ . □

Let  $\alpha' : F(A) \times B \rightarrow A$  be a  $\mathcal{K}$ -morphism and  $\beta : B \rightarrow F(B)$  be an  $F$ -coalgebra. A  $\mathcal{K}$ -morphism  $h : B \rightarrow A$  is a **para-hylo(morphism)** w.r.t.  $(\alpha', \beta)$  if

$$h = \alpha' \circ \langle F(h) \circ \beta, id_B \rangle \quad (8)$$

(see [73], section 5).

$\beta$  is **parametrically recursive** [9] if for every  $\mathcal{K}$ -morphism  $\alpha' : F(A) \times B \rightarrow A$  there is a unique para-hylo w.r.t.  $(\alpha', \beta)$ .

Since  $\langle F(h) \circ \beta, id_B \rangle = (F(h) \times id_B) \circ \langle \beta, id_B \rangle$ ,  $\alpha : F(A) \rightarrow A$  satisfies (5) iff

$$\alpha' = \alpha \circ \pi_1 : F(A) \times B \rightarrow A$$

solves (10), i.e., every parametrically recursive  $F$ -coalgebra is recursive.

Given a destructive signature  $\Sigma$ , the converse holds true as well: all recursive  $H_\Sigma$ -coalgebras (see chapter 13) are parametrically recursive ([9], Theorem 3.8).

Let  $\alpha : F(A) \rightarrow A$  be an  $F$ -algebra and  $\beta' : B \rightarrow F(B) + A$  be a  $\mathcal{K}$ -morphism. A  $\mathcal{K}$ -morphism  $h : B \rightarrow A$  is an **apo-hylo(morphism)** w.r.t.  $(\alpha, \beta')$  if

$$h = [\alpha \circ F(h), id_A] \circ \beta' \quad (9)$$

(see [73], section 5).

$\alpha$  is **parametrically corecursive** or **completely iterative** [8, 39] if for every  $\mathcal{K}$ -morphism  $\beta' : B \rightarrow F(B) + A$  there is a unique apo-hylo w.r.t.  $(\alpha, \beta')$ .

Since  $[\alpha \circ F(h), id_A] = [\alpha, id_A] \circ (F(h) + id_A)$ ,  $\beta : B \rightarrow F(B)$  satisfies (5) iff

$$\beta' = \beta \circ \iota_1 : B \rightarrow F(B) + A$$

solves (9), i.e., every parametrically corecursive  $F$ -algebra is corecursive.

According to [8], section 9, the converse does not hold true.

## Invariants and congruences on $F$ -(co)algebras

(see [77], Defs. 3.1.1, 3.1.2, 6.1.1, 6.2.1; [89], Def. 2.5)

Let  $F : \mathbf{Set} \rightarrow \mathbf{Set}$  be a functor,  $A$  be a set,  $B \subseteq A$  and  $R \subseteq A^2$ .

$$Pred(F)(B) =_{def} \{F(inc_B)(c) \mid c \in F(B)\} \subseteq F(A), \quad (\text{predicate lifting})$$

$$Rel(F)(R) =_{def} \{(F(\pi_1)(c), F(\pi_2)(c)) \mid c \in F(R)\} \subseteq F(A)^2. \quad (\text{relation lifting})$$

Let  $\alpha : F(A) \rightarrow A$  be an  $F$ -algebra,

$$\Phi_\alpha : \mathcal{P}(A) \rightarrow \mathcal{P}(A)$$

$$B \mapsto \{\alpha(c) \mid c \in \text{Pred}(F)(B)\},$$

$$\Psi_\alpha : \mathcal{P}(A^2) \rightarrow \mathcal{P}(A^2)$$

$$R \mapsto \{(\alpha(c), \alpha(d)) \mid (c, d) \in \text{Rel}(F)(R)\}.$$

$B$  is an **invariant** of  $\alpha$  if for all  $c \in \text{Pred}(F)(B)$ ,  $\alpha(c) \in B$ , or, equivalently, if  $B$  is  $\Phi_\alpha$ -closed.

$R$  is a **bisimulation** on  $\alpha$  if for all  $(c, d) \in \text{Rel}(F)(R)(B)$ ,  $(\alpha(c), \alpha(d)) \in R$ , or, equivalently, if  $R$  is  $\Psi_\alpha$ -closed.

By the **Fixpoint Theorem of Knaster and Tarski**,

$$\text{lfp}(\Phi_\alpha) = \bigcap \{B \subseteq A \mid B \text{ is } \Phi_\alpha\text{-closed}\},$$

$$\text{lfp}(\Psi_\alpha) = \bigcap \{R \subseteq A^2 \mid R \text{ is } \Psi_\alpha\text{-closed}\}.$$

Hence  $\text{lfp}(\Phi_\alpha)$  is the least invariant of  $\alpha$  and  $\text{lfp}(\Psi_\alpha)$  is the least bisimulation on  $\alpha$ .

Let  $\beta : A \rightarrow F(A)$  be an  $F$ -coalgebra,

$$\Phi_\beta : \mathcal{P}(A) \rightarrow \mathcal{P}(A)$$

$$B \mapsto \{a \in A \mid \beta(a) \in \text{Pred}(F)(B)\},$$

$$\Psi_\beta : \mathcal{P}(A^2) \rightarrow \mathcal{P}(A^2)$$

$$R \mapsto \{(a, b) \mid (\beta(a), \beta(b)) \in \text{Rel}(F)(R)\}.$$

$B$  is an **invariant** of  $\beta$  if for all  $a \in B$ ,  $\beta(a) \in \text{Pred}(F)(B)$ , or, equivalently, if  $B$  is  $\Phi_\beta$ -dense.

$R$  is a **bisimulation** on  $\beta$  if for all  $(a, b) \in R$ ,  $(\beta(a), \beta(b)) \in \text{Rel}(F)(R)$ , or, equivalently, if  $R$  is  $\Psi_\beta$ -dense.

By the **Fixpoint Theorem of Knaster and Tarski**,

$$gfp(\Phi_\beta) = \bigcup\{B \subseteq A \mid B \text{ is } \Phi_\beta\text{-dense}\},$$

$$gfp(\Psi_\beta) = \bigcup\{R \subseteq A^2 \mid R \text{ is } \Psi_\beta\text{-dense}\}.$$

Hence  $gfp(\Phi_\beta)$  is the greatest invariant of  $\beta$  and  $gfp(\Psi_\beta)$  is the greatest bisimulation on  $\beta$ . Moreover, by Theorem **BISIM** (2), greatest bisimulations are equivalence relations.

Therefore,  $gfp(\Psi_\beta)$  is also the greatest congruence on  $\beta$ .

## Complete categories and continuous functors

$\mathbb{O}$  denotes the category with ordinal numbers as objects and all pairs  $(i, j) \in \mathbb{O}^2$  with  $i \leq j$  as morphisms.

$\mathbb{O}_\lambda$  denotes the full subcategory of  $\mathbb{O}$  with all ordinal numbers less than  $\lambda$  as objects.

A **chain** of  $\mathcal{K}$  is a diagram  $\mathcal{D} : \mathbb{O} \rightarrow \mathcal{K}$ . A **cochain** of  $\mathcal{K}$  is a diagram  $\mathcal{D} : \mathbb{O} \rightarrow \mathcal{K}^{op}$ .

Let  $\lambda$  be an ordinal number.

A  **$\lambda$ -chain** of  $\mathcal{K}$  is a diagram  $\mathcal{D} : \mathbb{O}_\lambda \rightarrow \mathcal{K}$ . A  **$\lambda$ -cochain** of  $\mathcal{K}$  is a diagram  $\mathcal{D} : \mathbb{O}_\lambda \rightarrow \mathcal{K}^{op}$ .

$\mathcal{K}$  is  **$\lambda$ -cocomplete** if  $\mathcal{K}$  has an initial object and all  $\lambda$ -chains of  $\mathcal{K}$  have colimits.

$\mathcal{K}$  is  **$\lambda$ -complete** if  $\mathcal{K}$  has a final object and all  $\lambda$ -cochains of  $\mathcal{K}$  have limits.

Let  $\mathcal{K}$  and  $\mathcal{L}$  be  $\lambda$ -cocomplete. A functor  $F : \mathcal{K} \rightarrow \mathcal{L}$  is  **$\lambda$ -cocontinuous** if for all  $\lambda$ -chains  $\mathcal{D}$  of  $\mathcal{K}$ ,  $F$  preserves the colimit  $\{\mu_i : \mathcal{D}(i) \rightarrow C \mid i < \lambda\}$  of  $\mathcal{D}$ , i.e.,  $\{F(\mu_i) \mid i < \lambda\}$  is the colimit of  $F \circ \mathcal{D}$ .

Let  $\mathcal{K}$  and  $\mathcal{L}$  be  $\lambda$ -complete. A functor  $F : \mathcal{K} \rightarrow \mathcal{L}$  is  **$\lambda$ -continuous** if for all  $\lambda$ -cochains  $\mathcal{D}$  of  $\mathcal{K}$ ,  $F$  preserves the limit  $\{\nu_i : C \rightarrow \mathcal{D}(i) \mid i < \lambda\}$  of  $\mathcal{D}$ , i.e.,  $\{F(\nu_i) \mid i < \lambda\}$  is the limit of  $F \circ \mathcal{D}$ .

$CPO^E$  denotes the category of  $\omega$ -CPOs as objects and pairs

$$(f : A \rightarrow B, g : B \rightarrow A)$$

of  $\omega$ -continuous functions with  $g \circ f = id_A$  and  $f \circ g \leq id_B$  as morphisms.

**Theorem CPOE** (see, e.g., [115], section 11.3)

All endofunctors on  $CPO^E$  built up from identity and constant functors, coproducts, finite products and hom-functors are cocontinuous. □

$e \in \mathcal{T}_p(S, \mathcal{I})$  is **strongly polynomial** if  $e$  contains only product types with a *finite* set of indices.

Let  $\kappa$  be a cardinal number.  $e \in \mathcal{T}_p(S, \mathcal{I})$  is  **$\kappa$ -polynomial** if  $e$  does not contain a product type whose set of indices has a cardinality greater than  $\kappa$ .

## Theorem CONTYPES

(1) For all strongly polynomial types  $e$  over  $S$ ,  $F_e$  is  $\omega$ -continuous.

Let  $\kappa$  be a cardinal number and  $\lambda$  be the first **regular cardinal number**  $> \kappa$ . (For instance,  $\aleph_1 = |\cap \{\lambda \mid \lambda > \omega\}|$  is the first regular cardinal number  $> \omega$ .)

(2) For all  $\kappa$ -polynomial types  $e$  over  $S$ ,  $F_e$  is  $\lambda$ -cocontinuous.

(3) For all  $\kappa$ -polynomial types  $e$  over  $S$ ,  $F_e$  is  $\lambda$ -continuous.

*Proof.* By [17], Thms. 1 and 4, or [21], Prop. 2.2 (1) and (2), permutative and constant functors are  $\omega$ -continuous and  $\omega$ -cocontinuous,  $\omega$ -continuous or  $\lambda$ -cocontinuous functors are closed under coproducts,  $\omega$ -continuous functors are closed under products (and thus under exponentiation; see [143], Thm. 10.1) and  $\lambda$ -cocontinuous functors are closed under finite products.

By [21], Prop. 2.2 (3),  $\omega$ -continuous or  $\lambda$ -cocontinuous functors are closed under finite quotients, i.e., quotients consisting of *finite* equivalence classes. Since for all sets  $A$ ,  $A^* = \coprod_{n<\omega} A^n$  and  $\mathbb{N}_\omega^A \cong A^*/=_\text{bag}$ , the functors  $\underline{\phantom{x}}^*$  and  $\mathbb{N}_\omega^-$  are  $\omega$ -continuous and  $\omega$ -cocontinuous (see [10], Exs. 2.3.14/15).

By [10], Ex. 2.2.13,  $\mathcal{P}_\omega$  is  $\omega$ -cocontinuous. For a proof of the fact that  $\mathcal{P}_\omega$  is not  $\omega$ -continuous, see [10], Ex. 2.3.11.  $\mathcal{P}_\omega(A)$  is a quotient of  $A^*$ , but not a finite one:  $\mathcal{P}_\omega(A) \cong A^*/=_\text{set}$  (see [Preliminaries](#)).

By [10], Thm. 4.1.12,  $\lambda$ -cocontinuous functors are closed under products whose index sets have cardinalities less than  $\lambda$  and thus under exponentiation by exponents with a cardinality less than  $\lambda$ . Moreover,  $\omega$ -continuous or  $\lambda$ -cocontinuous functors are closed under sequential composition. □

## Categorical construction of initial $F$ -algebras and final $F$ -coalgebras

**Theorem INIALG** (For  $\lambda = \omega$ , see [15], section 2; [95], Thm. 2.1; for any  $\lambda$ , see [4], [6], Thm. 3.19, or [10], Cor. 4.1.5.)

Let  $\lambda$  be an infinite cardinal,  $Ini$  be initial in  $\mathcal{K}$  and  $\mathcal{K}$  be  $\kappa$ -cocomplete for all  $\kappa \leq \lambda$ .

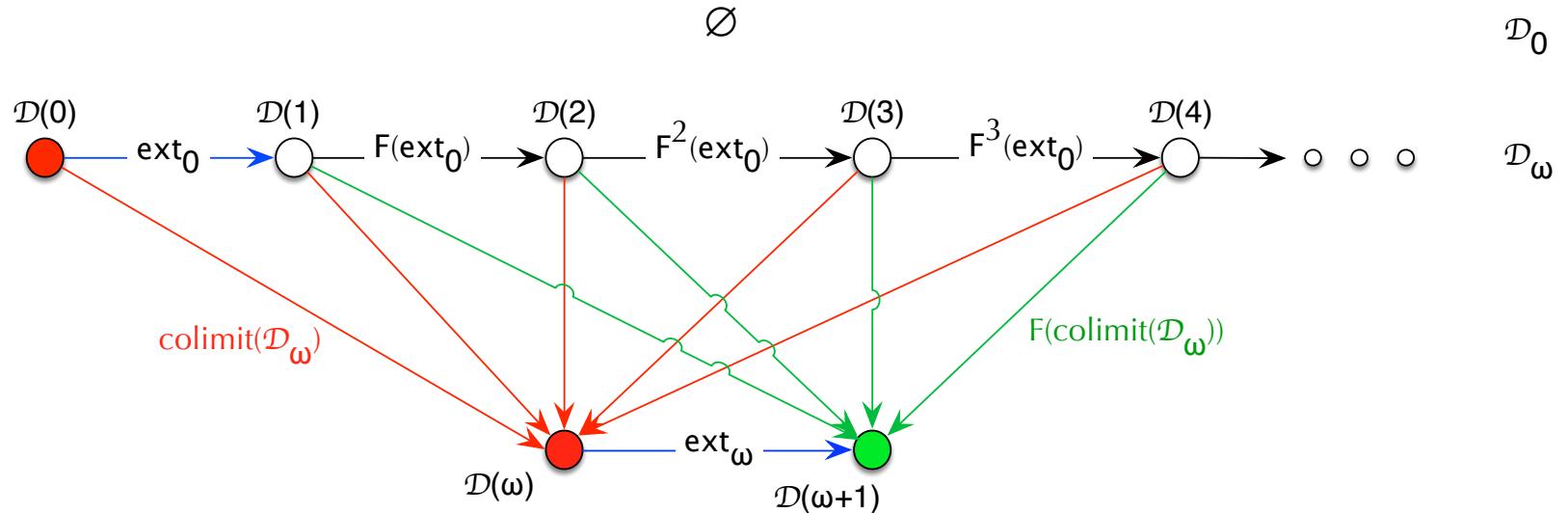
Given an endofunctor  $F$  on  $\mathcal{K}$ , define a  $\lambda$ -chain  $\mathcal{D}$  of  $\mathcal{K}$  as follows:

$$\begin{aligned}\mathcal{D}(0) &= Ini, \\ \mathcal{D}(0, 1) &= ext_0 : \mathcal{D}(0) \rightarrow \mathcal{D}(1), \\ \mathcal{D}(k+1) &= F(\mathcal{D}(k)) && \text{for all } k < \lambda, \\ \mathcal{D}(i+1, k+1) &= F(\mathcal{D}(i, k)) && \text{for all } i < k < \lambda, \\ \mathcal{D}(i, k) &= \mu_{i,k} : \mathcal{D}(i) \rightarrow \mathcal{D}(k) && \text{for all limit ordinals } k < \lambda \text{ and all } i < k, \\ \mathcal{D}(k, k+1) &= ext_k : \mathcal{D}(k) \rightarrow \mathcal{D}(k+1) && \text{for all limit ordinals } k < \lambda\end{aligned}$$

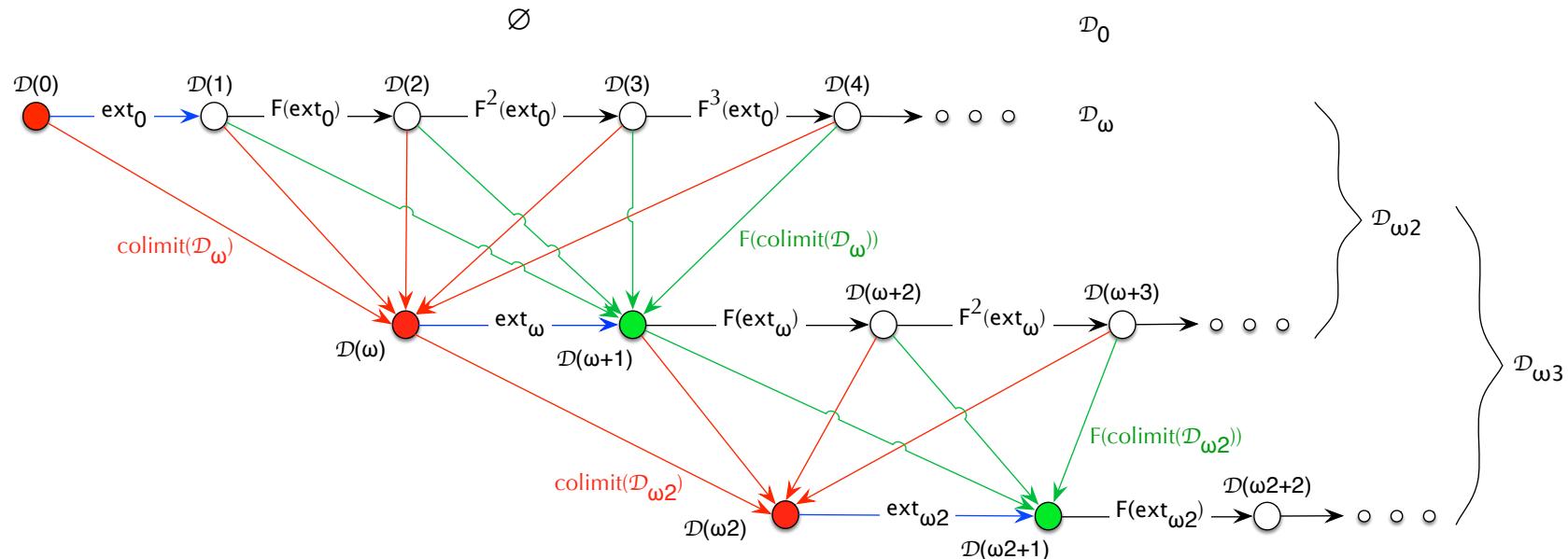
where  $ext_0$  is the unique  $\mathcal{K}$ -morphism from  $Ini$  to  $F(Ini)$  and for all limit ordinals  $k < \lambda$ ,  $\gamma_k = \{\mu_{i,k} \mid i < k\}$  is the colimit of the greatest subdiagram  $\mathcal{D}_k : \mathbb{O}_k \rightarrow \mathcal{K}$  of  $\mathcal{D}$  and  $ext_k$  is the unique  $\mathcal{K}$ -morphism from  $\mathcal{D}(k)$  to  $F(\mathcal{D}(k))$  such that for all  $i < k$ ,

$$ext_k \circ \mu_{i+1,k} = F(\mu_{i,k}) : \mathcal{D}(i+1) \rightarrow \mathcal{D}(k+1).$$

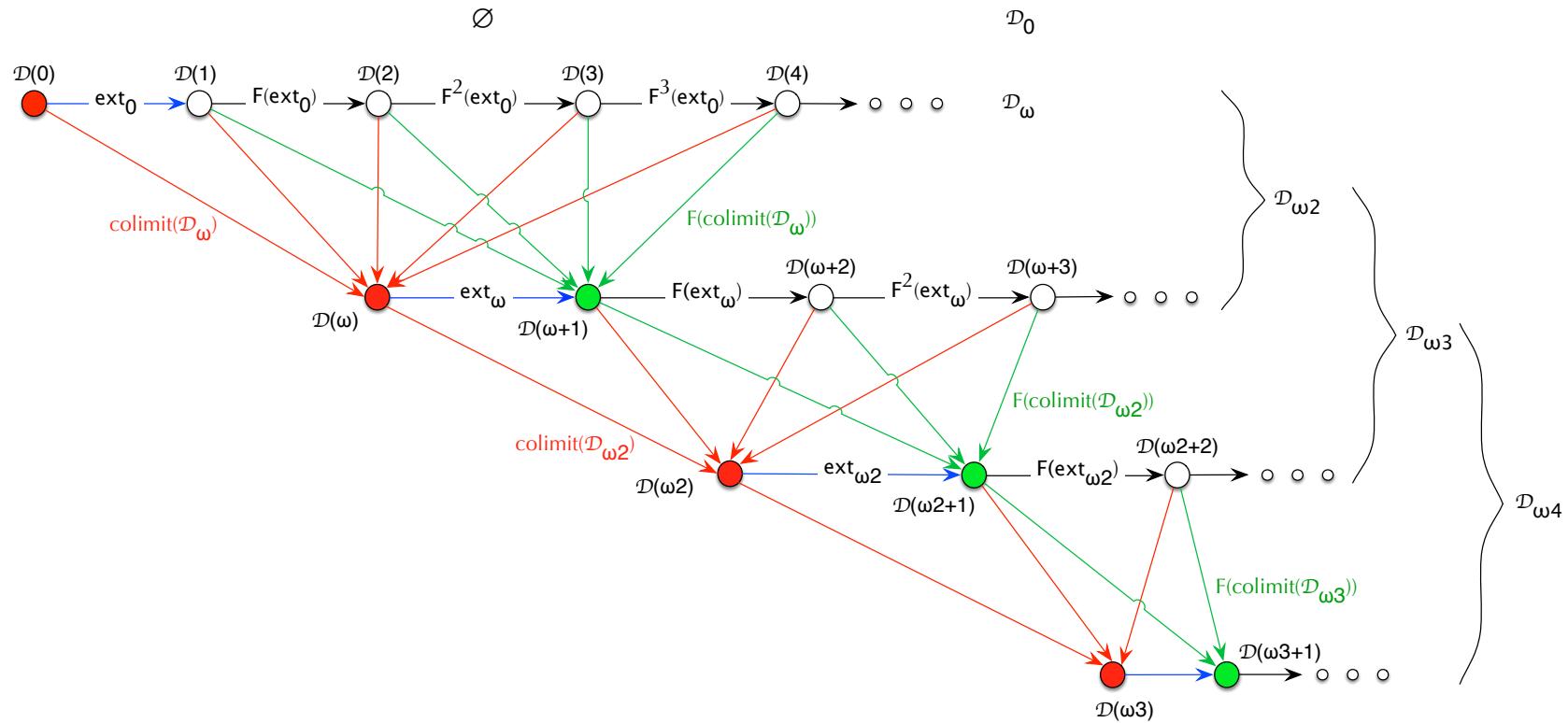
$ext_k$  exists because  $\{F(\mu_{i,k}) \mid i < k\}$  is a cocone of  $F \circ \mathcal{D}_k$  and  $\gamma_k \setminus \{\mu_{0,k}\}$  is the colimit of  $F \circ \mathcal{D}_k$ .



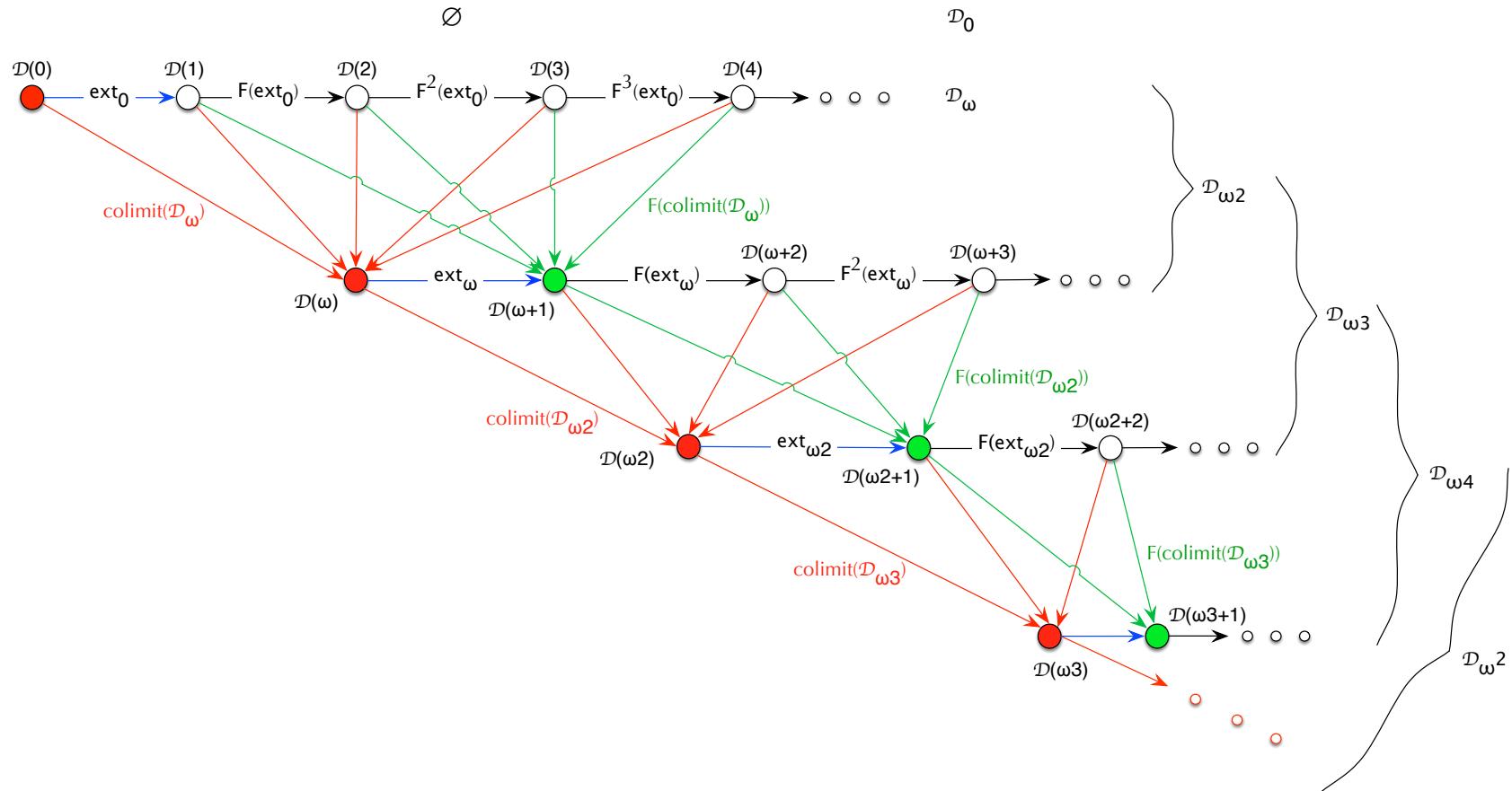
*The  $\omega + 2$ -chain of  $\mathcal{K}$  induced by the initial object  $\mathcal{D}(0)$  of  $\mathcal{K}$*



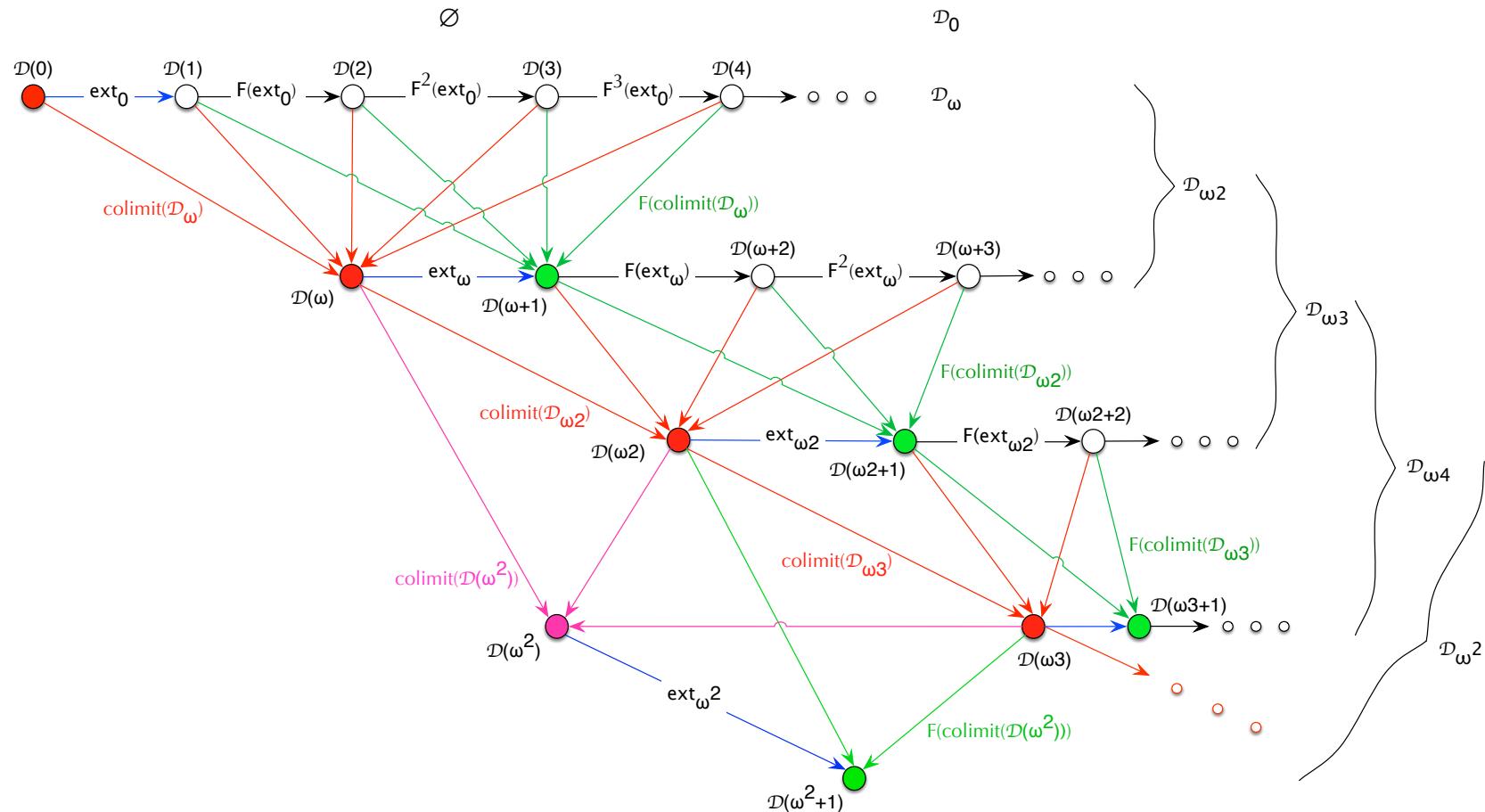
*The  $\omega_3$ -chain of  $\mathcal{K}$  induced by the initial object  $D(0)$  of  $\mathcal{K}$*



*The  $\omega_4$ -chain of  $\mathcal{K}$  induced by the initial object  $D(0)$  of  $\mathcal{K}$*



*The  $\omega^2$ -chain of  $\mathcal{K}$  induced by the initial object  $\mathcal{D}(0)$  of  $\mathcal{K}$*



The  $(\omega^2 + 2)$ -chain of  $\mathcal{K}$  induced by the initial object  $\mathcal{D}(0)$  of  $\mathcal{K}$

Let  $F$  be  $\lambda$ -cocontinuous,

$$\mu = \{\mu_i : \mathcal{D}(i) \rightarrow \mathcal{D}(\lambda) \mid i < \lambda\}$$

be the colimit of  $\mathcal{D}$ . Then

$$F(\mu) = \{F(\mu_i) : F(\mathcal{D}(i)) \rightarrow F(\mathcal{D}(\lambda)) \mid i < \lambda\}$$

is the colimit of  $F \circ \mathcal{D}$ . Since  $\mu \setminus \{\mu_0\}$  is a cocone of  $F \circ \mathcal{D}$ , there is a unique  $\mathcal{K}$ -morphism  $ext : F(\mathcal{D}(\lambda)) \rightarrow \mathcal{D}(\lambda)$ —and thus an  $F$ -algebra—such that for all  $i < \lambda$ ,

$$ext \circ F(\mu_i) = \mu_{i+1} : \mathcal{D}(i+1) \rightarrow \mathcal{D}(\lambda).$$

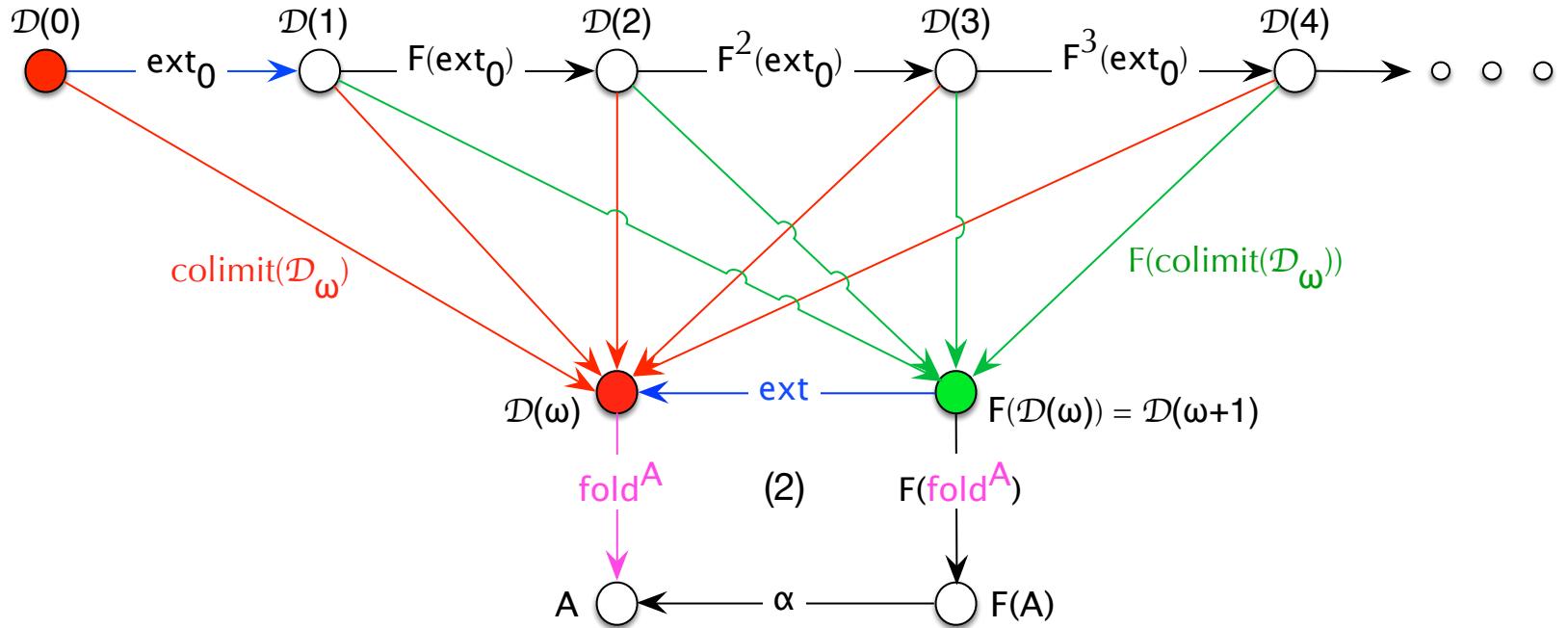
*ext* is initial in  $Alg_F$  and thus, by Lambek's Lemma (1),  $F(\mathcal{D}(\lambda)) \cong \mathcal{D}(\lambda)$ .

*Proof.* Let  $\alpha : F(A) \rightarrow A$  be an  $F$ -algebra. Since  $\mathcal{D}(0) = Ini$  is initial in  $\mathcal{K}$ , there is a unique  $\mathcal{K}$ -morphism  $ini^A$  from  $Ini$  to  $A$ . Hence  $\mathcal{D}$  has the cocone

$$\nu = \{\nu_i : \mathcal{D}(i) \rightarrow A \mid i < \lambda\}$$

with  $\nu_0 = ini^A$  and  $\nu_{i+1} = \alpha \circ F(\nu_i)$  for all  $i < \lambda$ . We obtain a unique  $\mathcal{K}$ -morphism  $fold^A : \mathcal{D}(\lambda) \rightarrow A$  with  $fold^A \circ \mu_i = \nu_i$  for all  $i < \lambda$ . Therefore,

$$\begin{aligned} fold^A \circ ext \circ F(\mu_i) &= fold^A \circ \mu_{i+1} = \nu_{i+1} = \alpha \circ F(\nu_i) = \alpha \circ F(fold^A \circ \mu_i) \\ &= \alpha \circ F(fold^A) \circ F(\mu_i). \end{aligned} \tag{1}$$



The initial  $F$ -algebra  $\text{ext}$  in the case  $\lambda = \omega$

Since  $\nu \setminus \{\nu_0\}$  is a cocone of  $F \circ \mathcal{D}$ —with target  $A$ —and  $\mu \setminus \{\mu_0\}$  is the colimit of  $F \circ \mathcal{D}$ —with target  $F(\mathcal{D}(\lambda))$ —, there is only one  $\mathcal{K}$ -morphism  $h : F(\mathcal{D}(\lambda)) \rightarrow A$  with  $h \circ F(\mu_i) = \nu_{i+1}$  for all  $i < \lambda$ . Hence (1) implies

$$\text{fold}^A \circ \text{ext} = \alpha \circ F(\text{fold}^A), \quad (2)$$

i.e.,  $\text{fold}^A$  is an  $\text{Alg}_F$ -morphism from  $\text{ext}$  to  $\alpha$ .

It remains to show that  $\text{fold}^A$  is the only  $\text{Alg}_F$ -morphism from  $\text{ext}$  to  $\alpha$ .

Let  $\theta : \mathcal{D}(\lambda) \rightarrow A$  be an  $\text{Alg}_F$ -morphism from  $\text{ext}$  to  $\alpha$ , i.e.,

$$\theta \circ \text{ext} = \alpha \circ F(\theta). \quad (3)$$

Suppose that for all  $i < \lambda$ ,

$$\theta \circ \mu_i = \nu_i : \mathcal{D}(i) \rightarrow A. \quad (4)$$

Since  $\text{fold}^A \circ \mu_i = \nu_i$  and there is only one  $\mathcal{K}$ -morphism  $h : \mathcal{D}(\lambda) \rightarrow A$  with  $h \circ \mu_i = \nu_i$ , we conclude  $\theta = \text{fold}^A$ .

It remains to show (4) by transfinite induction on  $i$ .

Since  $\mathcal{D}(0) = \text{Ini}$  is initial in  $\mathcal{K}$ ,  $\theta \circ \mu_0 = \nu_0$ . Let  $0 < k < \lambda$ .

If  $k$  is a successor ordinal, then  $k = i + 1$  for some ordinal  $i$  and thus

$$\begin{aligned} \theta \circ \mu_k &= \theta \circ \mu_{i+1} = \theta \circ \text{ext} \circ F(\mu_i) \stackrel{(3)}{=} \alpha \circ F(\theta) \circ F(\mu_i) = \alpha \circ F(\theta \circ \mu_i) \\ &\stackrel{\text{ind. hyp.}}{=} \alpha \circ F(\nu_i) = \nu_{i+1} = \nu_k. \end{aligned}$$

Let  $k$  be a limit ordinal. Since  $\mu$  and  $\nu$  are cocones of  $\mathcal{D}$ ,  $\mu_k \circ \mu_{i,k} = \mu_i$  and  $\nu_k \circ \mu_{i,k} = \nu_i$  for all  $i \in k$ . Again by induction hypothesis,

$$\theta \circ \mu_k \circ \mu_{i,k} = \theta \circ \mu_i = \nu_i = \nu_k \circ \mu_{i,k}. \quad (5)$$

Since  $\{\nu_i \mid i < k\}$  is a cocone of  $\mathcal{D}_k$ —with target  $A$ —and  $\{\mu_{i,k} \mid i < k\}$  is the colimit of  $\mathcal{D}_k$ —with target  $\mathcal{D}(k)$ —, there is only one  $\mathcal{K}$ -morphism  $h : \mathcal{D}(k) \rightarrow A$  with  $h \circ \mu_{i,k} = \nu_i$  for all  $i < k$ . Hence (5) implies  $\theta \circ \mu_k = \nu_k$ , and the proof of (4) is complete.  $\square$

## Theorem FINALG

Let  $\lambda$  be an infinite cardinal,  $Fin$  be final in  $\mathcal{K}$  and  $\mathcal{K}$  be  $\kappa$ -complete for all  $\kappa \leq \lambda$ .

Given an endofunctor  $F$  on  $\mathcal{K}$ , define a  $\lambda$ -cochain  $\mathcal{D}$  of  $\mathcal{K}$  as follows:

$$\begin{aligned}\mathcal{D}(0) &= Fin, \\ \mathcal{D}(1, 0) &= ext_0 : \mathcal{D}(1) \rightarrow \mathcal{D}(0), \\ \mathcal{D}(k+1) &= F(\mathcal{D}(k)) && \text{for all } k < \lambda, \\ \mathcal{D}(k+1, i+1) &= F(\mathcal{D}(k, i)) && \text{for all } i < k < \lambda, \\ \mathcal{D}(k, i) &= \mu_{k,i} : \mathcal{D}(k) \rightarrow \mathcal{D}(i) && \text{for all limit ordinals } k < \lambda \text{ and all } i < k, \\ \mathcal{D}(k+1, k) &= ext_k : \mathcal{D}(k+1) \rightarrow \mathcal{D}(k) && \text{for all limit ordinals } k < \lambda\end{aligned}$$

where  $ext_0$  is the unique  $\mathcal{K}$ -morphism from  $F(Fin)$  to  $Fin$  and for all limit ordinals  $k < \lambda$ ,  $\gamma_k = \{\mu_{k,i} \mid i < k\}$  is the limit of the greatest subdiagram  $\mathcal{D}_k : \mathbb{O}_k \rightarrow \mathcal{K}$  of  $\mathcal{D}$  and  $ext_k$  is the unique  $\mathcal{K}$ -morphism from  $F(\mathcal{D}(k))$  to  $\mathcal{D}(k)$  such that for all  $i < k$ ,

$$\mu_{k,i+1} \circ ext_k = F(\mu_{k,i}) : \mathcal{D}(k+1) \rightarrow \mathcal{D}(i+1).$$

$ext_k$  exists because  $\{F(\mu_{k,i}) \mid i < k\}$  is a cone of  $F \circ \mathcal{D}_k$  and  $\gamma_k \setminus \{\mu_{k,0}\}$  is the limit of  $F \circ \mathcal{D}_k$ .

Let  $F$  be  $\lambda$ -continuous,

$$\mu = \{\mu_i : \mathcal{D}(\lambda) \rightarrow \mathcal{D}(i) \mid i < \lambda\}$$

be the limit of  $\mathcal{D}$ . Then

$$F(\mu) = \{F(\mu_i) : F(\mathcal{D}(\lambda)) \rightarrow F(\mathcal{D}(i)) \mid i < \lambda\}$$

is the limit of  $F \circ \mathcal{D}$ . Since  $\mu \setminus \{\mu_0\}$  is a cone of  $F \circ \mathcal{D}$ , there is a unique  $\mathcal{K}$ -morphism  $ext : \mathcal{D}(\lambda) \rightarrow F(\mathcal{D}(\lambda))$ —and thus an  $F$ -coalgebra—such that for all  $i < \lambda$ ,

$$F(\mu_i) \circ ext = \mu_{i+1} : \mathcal{D}(\lambda) \rightarrow \mathcal{D}(i+1).$$

$ext$  is final in  $coAlg_F$  and thus, by **Lambek's Lemma (2)**,  $\mathcal{D}(\lambda) \cong F(\mathcal{D}(\lambda))$ .

*Proof.* Let  $\alpha : A \rightarrow F(A)$  be an  $F$ -coalgebra. Since  $\mathcal{D}(0) = Fin$  is final in  $\mathcal{K}$ , there is a unique  $\mathcal{K}$ -morphism  $fin^A$  from  $A$  to  $\mathcal{D}(0)$ . Hence  $\mathcal{D}$  has the cone

$$\nu = \{\nu_i : A \rightarrow \mathcal{D}(i) \mid i < \lambda\}$$

with  $\nu_0 = fin^A$  and  $\nu_{i+1} = F(\nu_i) \circ \alpha$  for all  $i < \lambda$ . We obtain a unique  $\mathcal{K}$ -morphism  $unfold^A : A \rightarrow \mathcal{D}(\lambda)$  with  $\mu_i \circ unfold^A = \nu_i$  for all  $i < \lambda$ .

Proceed analogously to the proof of Theorem INIALG. □

## Corollary

Suppose that all (co)chains of  $\mathcal{K}$  have (co)limits. Then the definition of the  $\lambda$ -(co)chain  $\mathcal{D}$  in Theorem INIALG or FINALG can be extended to the definition of a (co)chain.

If  $F : \mathcal{K} \rightarrow \mathcal{K}$  is  $\lambda$ -(co)continuous, then  $\mathcal{D}$  **converges in  $\lambda$  steps**, i.e.,  $\mathcal{D}(\lambda) \cong \mathcal{D}(\lambda + 1)$ .

*Proof.* The conjecture follows immediately from [Lambek's Lemma](#) and Theorem INIALG or FINALG.  $\square$

## Functors for constructive signatures

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive signature.

$\Sigma$  induces the functor  $H_\Sigma : Mod(S, \mathcal{I}) \rightarrow Mod(S, \mathcal{I})$  that is defined as follows:

For all  $A, B \in Mod(S, \mathcal{I})$ ,  $Mod(S, \mathcal{I})$ -morphisms  $h : A \rightarrow B$  and  $s \in S$ ,

$$\begin{aligned} H_\Sigma(A)_s &= \coprod_{c:e \rightarrow s \in C} A_e, \\ H_\Sigma(h)_s &= \coprod_{c:e \rightarrow s \in C} h_e. \end{aligned}$$

An  $H_\Sigma$ -algebra  $\alpha : H_\Sigma(A) \rightarrow A$  (see  $F$ -algebras and  $F$ -coalgebras) uniquely corresponds to a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  and vice versa:

For all  $s \in S$  and  $c : e \rightarrow s \in C$ ,

$$\begin{array}{ccc} H_\Sigma(A)_s & \xrightarrow{\alpha_s = [c^{\mathcal{A}}]_{c:e \rightarrow s \in C}} & A_s \\ \uparrow \iota_c & \nearrow (1) & \\ A_e & \xrightarrow{c^{\mathcal{A}} = \alpha_s \circ \iota_c} & \end{array}$$

Hence  $\alpha_s$  is the sum extension of the interpretations of all constructors of  $\Sigma$  in  $A$ .

Moreover, given  $\Sigma$ -algebras  $\mathcal{A}, \mathcal{B}$  and corresponding  $H_\Sigma$ -algebras  $\alpha, \beta$ , an  $S$ -sorted function  $h : A \rightarrow B$  is  $\Sigma$ -homomorphic iff  $h$  is an  $Alg_{H_\Sigma}$ -morphism from  $\alpha$  to  $\beta$ .

A  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  is initial in  $Alg_\Sigma$  iff the corresponding  $H_\Sigma$ -algebra  $\alpha : H_\Sigma(A) \rightarrow A$  is initial in  $Alg_{H_\Sigma}$ .

Hence by **Lambek's Lemma** (1), if  $\mathcal{A}$  is initial in  $Alg_\Sigma$ , then  $\alpha$  is iso and thus

- $A_s$  is a sum of  $(A_e)_{c:e \rightarrow s \in C}$  with injections  $c^{\mathcal{A}} : A_e \rightarrow A_s$ ,
- for all  $\lambda\Sigma$ -term tuples  $(t_c : e \rightarrow e_s)_{c:e \rightarrow s \in C}$ , the **case distinction**

$$\text{case}\{c.t_c\}_{c:e \rightarrow s \in C} : s \rightarrow e_s$$

has a well-defined interpretation in  $\mathcal{A}$ : For all  $g \in hval(V, A)$ , the inductive definition of  $g^* : \Lambda_\Sigma(V) \rightarrow A$  is extended by the equation

$$g^*(\text{case}\{c.t_c\}_{c:e \rightarrow s \in C}) = [g^*(t_c)]_{c:e \rightarrow s \in C} \circ [c^{\mathcal{A}}]_{c:e \rightarrow s \in C}^{-1}$$

(see  **$\Sigma$ -formulas**).

Case distinctions as defined above are functional versions of **case**-statements and variant types in the sense of [63] and [1], respectively.

By (1) and Lemma ISO (2), for all  $c : e \rightarrow s \in C$ ,  $\iota_c = \alpha_s^{-1} \circ c^{\mathcal{A}}$  and thus

$$g^*(\text{case}\{c.t_c\}_{c:e \rightarrow s \in C}) \circ c^{\mathcal{A}} = [g^*(t_c)]_{c:e \rightarrow s \in C} \circ \alpha_s^{-1} \circ c^{\mathcal{A}} = [g^*(t_c)]_{c:e \rightarrow s \in C} \circ \iota_c = g^*(t_c).$$

Conversely, let  $d = (d_s : A_s \rightarrow A_{e_s})_{s \in S}$  be an  $S$ -sorted function such that for all  $c : e \rightarrow s \in C$ ,  $d_s \circ c^{\mathcal{A}} = t_c^{\mathcal{A}}$  for some  $t_c \in cl \in \Lambda_{\Sigma}(V)$ . Then by (1),

$$d_s \circ \alpha_s \circ \iota_c = d_s \circ c^{\mathcal{A}} = t_c^{\mathcal{A}} = [t_c^{\mathcal{A}}]_{c:e \rightarrow s \in C} \circ \iota_c$$

and thus  $d_s \circ \alpha_s = [t_c^{\mathcal{A}}]_{c:e \rightarrow s \in C}$ . Hence

$$d_s = d_s \circ \alpha_s \circ \alpha_s^{-1} = [t_c^{\mathcal{A}}]_{c:e \rightarrow s \in C} \circ \alpha_s^{-1} = (\text{case}\{c.t_c\}_{c:e \rightarrow s \in C})^{\mathcal{A}}.$$

## Examples

Let  $A$  be an  $S$ -sorted set and  $I, X, Y, Act$  be as in chapter 11. We omit sort indices if  $S$  is a singleton.

$$H_{Mon}(A) = 1 + A \times A,$$

$$H_{Nat}(A) = 1 + A,$$

$$H_{Dyn(X,Y)}(A) = X \times A + Y,$$

$$H_{coStream(X)}(A) = X \times A,$$

$$H_{Bintree(X)}(A) = X \times A \times A + 1,$$

$$H_{Tree(X)}(A) = X \times A^*,$$

$$H_{Reg(X)}(A) = A^2 + A^2 + A + 1 + \mathcal{P}(X),$$

$$H_{CCS(Act)}(A) = Act + A^2 + A^2 + A \times Act + A \times Act^{Act}. \quad \square$$

## Functors for destructive signatures

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive signature.

$\Sigma$  induces the functor  $H_\Sigma : Mod(S, \mathcal{I}) \rightarrow Mod(S, \mathcal{I})$ : For all  $A, B \in Mod(S, \mathcal{I})$ ,  $Mod(S, \mathcal{I})$ -morphisms  $h : A \rightarrow B$  and  $s \in S$ ,

$$\begin{aligned} H_\Sigma(A)_s &= \prod_{d:s \rightarrow e \in D} A_e, \\ H_\Sigma(h)_s &= \prod_{d:s \rightarrow e \in D} h_e. \end{aligned}$$

An  $H_\Sigma$ -coalgebra  $\alpha : A \rightarrow H_\Sigma(A)$  (see  $F$ -algebras and  $F$ -coalgebras) uniquely corresponds to a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  and vice versa:

For all  $s \in S$  and  $d : s \rightarrow e \in D$ ,

$$\begin{array}{ccc} A_s & \xrightarrow{\alpha_s = \langle d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}} & H_\Sigma(A)_s \\ & \searrow d^{\mathcal{A}} = \pi_d \circ \alpha_s & \downarrow \pi_d \\ & & A_e \end{array}$$

(2)

Hence  $\alpha_s$  is the product extension of the interpretations of all destructors of  $\Sigma$  in  $A$ .

Moreover, given  $\Sigma$ -algebras  $\mathcal{A}, \mathcal{B}$  and corresponding  $H_\Sigma$ -coalgebras  $\alpha, \beta$ , an  $S$ -sorted function  $h : A \rightarrow B$  is  $\Sigma$ -homomorphic iff  $h$  is a  $coAlg_{H_\Sigma}$ -morphism from  $\alpha$  to  $\beta$ .

A  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  is final in  $Alg_\Sigma$  iff the corresponding  $H_\Sigma$ -coalgebra  $\alpha : A \rightarrow H_\Sigma(A)$  is final in  $coAlg_{H_\Sigma}$ .

Hence by Lambek's Lemma (2),  $\alpha$  is iso and thus

- $A_s$  is a product of  $(A_e)_{d:s \rightarrow e \in D}$  with projections  $d^{\mathcal{A}} : A_s \rightarrow A_e$ ,
- for all  $\lambda\Sigma$ -Term tuples  $(t_d : E_s \rightarrow e)_{d:s \rightarrow e}$ , the **object definition**

$$obj\{d.t_d\}_{d:s \rightarrow e \in D} : e_s \rightarrow s$$

has a well-defined interpretation in  $\mathcal{A}$ : For all  $g \in hval(V, A)$ , the inductive definition of  $g^* : \Lambda_\Sigma(V) \rightarrow A$  is extended by the equation

$$g^*(obj\{d.t_d\}_{d:s \rightarrow e \in D}) = \langle d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}^{-1} \circ \langle g^*(t_d) \rangle_{d:s \rightarrow e \in D}$$

(see  $\Sigma$ -formulas).

Object definitions as defined above are functional versions of `merge`-statements and record types in the sense of [63] and [1], respectively.

By (2) and Lemma ISO (1), for all  $d : s \rightarrow e \in D$ ,  $\pi_d = d^{\mathcal{A}} \circ \alpha_s^{-1}$  and thus

$$d^{\mathcal{A}} \circ g^*(\text{obj}\{d.t_d\}_{d:s \rightarrow e \in D}) = d^{\mathcal{A}} \circ \alpha_s^{-1} \circ \langle g^*(t_d) \rangle_{d:s \rightarrow e \in D} = \pi_d \circ \langle g^*(t_d) \rangle_{d:s \rightarrow e \in D} = g^*(t_d).$$

Conversely, let  $c = (c_s : A_{e_s} \rightarrow A_s)_{s \in S}$  be an  $S$ -sorted function such that for all  $d : s \rightarrow e \in D$ ,  $d^{\mathcal{A}} \circ c_s = t_d^{\mathcal{A}}$  for some  $t_d \in \text{cl}\Lambda_{\Sigma}(V)$ . Then by (2),

$$\pi_d \circ \alpha_s \circ c_s = d^{\mathcal{A}} \circ c_s = t_d^{\mathcal{A}} = \pi_d \circ \langle t_d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}$$

and thus  $\alpha_s \circ c_s = \langle t_d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}$ . Hence

$$c = \alpha_s^{-1} \circ \alpha_s \circ c_s = \alpha_s^{-1} \circ \langle t_d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D} = (\text{obj}\{d.t_d\}_{d:s \rightarrow e \in D})^{\mathcal{A}}.$$

## Examples

Let  $A$  be an  $S$ -sorted set and  $X, Y, \text{Act}, M, \varphi, \Sigma = (S, \mathcal{I}, C)$  be as in chapter 11. We omit sort indices if  $S$  is a singleton.

$$\begin{aligned} H_{coNat}(A) &= 1 + A, \\ H_{Stream(X)}(A) &= X \times A, \\ H_{coDyn(X,Y)}(A) &= X \times A + Y, \\ H_{infBintree(X)}(A) &= A \times X \times A, \\ H_{coBintree(X)}(A) &= X \times A \times A + 1, \end{aligned}$$

$$\begin{aligned}
 H_{infTree(X)}(A) &= X \times A^+, \\
 H_{coTree_\omega(X)}(A) &= X \times A^*, \\
 H_{coTree(X)}(A)_{tree} &= X \times A_{trees}, \\
 H_{coTree(X)}(A)_{trees} &= A_{tree} \times A_{trees} + 1, \\
 H_{Proctree(Act)}(A) &= (Act \times A)^*, \\
 H_{WStream(X,M)}(A) &= X \times M_\omega^A, \\
 H_{WStream^*(X,M)}(A) &= X \times (A \times M)^*, \\
 H_{Med(X)}(A) &= A^X, \\
 H_{NMed(X)}(A) &= \mathcal{P}_\omega(A)^X, \\
 H_{NMed^*(X)}(A) &= (A^*)^X, \\
 H_{WMed(X,M)}(A) &= (M_\omega^A)^X, \\
 H_{WMed^*(X,M)}(A) &= ((M \times A)^*)^X, \\
 H_{DAut(X,Y)}(A) &= A^X \times Y, \\
 H_{Mealy(X,Y)}(A) &= A^X \times Y^X, \\
 H_{PAut(X,Y)}(A) &= (1 + A)^X \times Y, \\
 H_{NAut(X,Y)}(A) &= \mathcal{P}_\omega(A)^X \times Y, \\
 H_{NAut^*(X,Y)}(A) &= (A^*)^X \times Y,
 \end{aligned}$$

$$\begin{aligned}
 H_{WAut(X,M,Y)}(A) &= (M_\omega^A)^X \times Y, \\
 H_{WAut^*(X,M,Y)}(A) &= ((A \times M)^*)^X \times Y, \\
 H_{PrAut(X,Y)}(A) &= \mathcal{D}_\omega(A)^X \times Y, \\
 H_{TAcc(\Sigma)}(A)_s &= \prod_{c:e \rightarrow s \in C} A_e, \quad s \in S, \\
 H_{NTAcc(\Sigma)}(A)_s &= \prod_{c:e \rightarrow s \in C} \mathcal{P}_\omega(A_e), \quad s \in S, \\
 H_{NTAcc^*(\Sigma)}(A)_s &= \prod_{c:e \rightarrow s \in C} A_e^*, \quad s \in S, \\
 H_{Class(BS)}(A) &= \prod_{i=1}^n ((Y_i \times A) + E_i)^{X_i}, \\
 H_{Graph(X,Y)}(A)_{node} &= X, \\
 H_{Graph(X,Y)}(A)_{edge} &= A_{node} \times A_{node} \times Y. \quad \square
 \end{aligned}$$

Let  $\Sigma$  and  $\Sigma'$  be both constructive or both destructive signatures with bijective sets of sorts.

$\Sigma$  and  $\Sigma'$  are **equivalent** if  $H_\Sigma$  and  $H_{\Sigma'}$  are naturally equivalent (modulo the bijection on sort sets).

$\Sigma'$  is a **quotient** of  $\Sigma$  if there is a surjective natural transformation from  $H_\Sigma$  to  $H_{\Sigma'}$ .

## Final models of destructive non-polynomial signatures

**Lemma WEAKFIN** (see [10], 2.4.6/16; [61], 4.3.2/3)

Let  $\Sigma = (S, \mathcal{I}, D)$  and  $\Sigma' = (S, \mathcal{I}, D')$  be destructive signatures,  $\tau : H_\Sigma \rightarrow H_{\Sigma'}$  be a surjective natural transformation,  $\mathcal{A}$  be final in  $Alg_\Sigma$  and  $\alpha : A \rightarrow H_\Sigma(A)$  be the corresponding  $H_\Sigma$ -coalgebra where  $A$  is the carrier of  $\mathcal{A}$  (see (2)).

Then  $\tau_A \circ \alpha : A \rightarrow H_{\Sigma'}(A)$  is a **weakly final**  $H_{\Sigma'}$ -coalgebra, i.e., for every  $H_{\Sigma'}$ -coalgebra  $\beta$  there is a  $coAlg_{H_{\Sigma'}}$ -morphism from  $\beta$  to  $\tau_A \circ \alpha$ .

Moreover,  $\mathcal{A}/\sim$  is final in  $Alg_{\Sigma'}$  where  $\sim$  is the greatest  $\Sigma$ -bisimulation on  $A$  (which, by Theorem BISIM (2), is a  $\Sigma$ -congruence) and for all  $d \in D'$ ,  $d^{\mathcal{A}/\sim} = \pi_d \circ \tau_A \circ \alpha/\sim$ .

*Proof.*

Let  $\beta : B \rightarrow H_{\Sigma'}(B)$  be a  $H_{\Sigma'}$ -coalgebra (see (1)). Since  $\tau_B : H_\Sigma(B) \rightarrow H_{\Sigma'}(B)$  is surjective, there is an  $S$ -sorted function  $h : H_{\Sigma'}(B) \rightarrow H_\Sigma(B)$  with  $\tau_B \circ h = id_{H_{\Sigma'}(B)}$ .

Hence  $h \circ \beta : B \rightarrow H_\Sigma(B)$  is a  $H_\Sigma$ -coalgebra and thus there is a unique  $\Sigma$ -homomorphism  $unfold^B : B \rightarrow A$  from  $h \circ \beta$  to  $\alpha$ .

$unfold^B$  is also a  $\Sigma'$ -homomorphism from  $\beta$  to  $\tau_A \circ \alpha : A \rightarrow H_{\Sigma'}(A)$ :

$$H_{\Sigma'}(unfold^B) \circ \beta = H_{\Sigma'}(unfold^B) \circ \tau_B \circ h \circ \beta \stackrel{\tau \text{ natural}}{=} \tau_A \circ H_{\Sigma}(unfold^B) \circ h \circ \beta \\ unfold^B \stackrel{\Sigma\text{-hom.}}{=} \tau_A \circ \alpha \circ unfold^B.$$

Hence  $nat_{\sim} \circ unfold^B : B \rightarrow A/\sim$  is a  $\Sigma'$ -homomorphism from  $\beta$  to  $\tau_A \circ \alpha/\sim$ .

It is unique: Let  $f, g : B \rightarrow A/\sim$  be  $\Sigma'$ -homomorphisms from  $\beta$  to  $\tau_A \circ \alpha/\sim$ . Then there is an  $S$ -sorted function  $h : A/\sim \rightarrow A$  with  $nat_{\sim} \circ h = id_{A/\sim}$ .

Let  $\approx$  be the least  $\Sigma$ -congruence on  $A$  that contains all pairs  $(h(f(b)), h(g(b)))$  with  $b \in B$ . Since  $\sim$  is the largest  $\Sigma$ -congruence on  $A$ ,  $\approx \subseteq \sim$ .

Hence for all  $b \in B$ ,  $h(f(b)) \approx h(g(b))$  implies  $h(f(b)) \sim h(g(b))$  and thus

$$f(b) = nat_{\sim}(h(f(b))) = nat_{\sim}(h(g(b))) = g(b).$$

Therefore,  $g = h$ . We conclude that  $\tau_A \circ \alpha/\sim$  is final in  $coAlg_{H_{\Sigma'}}$  and thus  $\mathcal{A}/\sim$  is final in  $Alg_{\Sigma'}$ . □

Examples Given sets  $X, Y$  and a commutative monoid  $M$ , let the mappings

$$\begin{aligned}\tau_1 : H_{WStream^*(X,M)} &\rightarrow H_{WStream(X,M)}, \\ \tau_2 : H_{NMed^*(X)} &\rightarrow H_{NMed(X)}, \\ \tau_3 : H_{WMed^*(X,M)} &\rightarrow H_{WMed(X,M)}, \\ \tau_4 : H_{NAut^*(X,Y)} &\rightarrow H_{NAut(X,Y)}, \\ \tau_5 : H_{WAut^*(X,M,Y)} &\rightarrow H_{WAut(X,M,Y)}, \\ \tau_6 : H_{WAut^*(X,\mathbb{R}_{\geq 0},Y)} &\rightarrow H_{PrAut(X,Y)}, \\ \tau_7 : H_{NTAcc^*(\Sigma)} &\rightarrow H_{NTAcc(\Sigma)}\end{aligned}$$

be defined as follows: Let  $A$  be a set.

- For all  $(x, ps) \in X \times (A \times M)^* = H_{WStream^*(X,M)}(A)$ ,

$$\tau_{1,A}(x, ps) = (x, \lambda a. \sum_{(a,m) \in ps} m) \in X \times M_\omega^A = H_{WStream(X,M)}(A).$$

- For all  $f \in (A^*)^X = H_{NMed^*(X)}(A)$ ,

$$\tau_{2,A}(f) = \lambda x. \{\pi_i(f(x)) \mid 1 \leq i \leq |f(x)|\} \in \mathcal{P}_\omega(A)^X = H_{NMed(X)}(A).$$

- For all  $f \in ((M \times A)^*)^X = H_{WMed^*(X)}(A)$ ,

$$\tau_{3,A}(f) = \lambda x. \lambda a. \sum_{(a,m) \in f(x)} r \in (M_\omega^A)^X = H_{WMed(X,M)}(A).$$

- For all  $(f, y) \in (A^*)^X \times Y = H_{NAut^*(X,Y)}(A)$ ,

$$\tau_{4,A}(f, y) = (\tau_{2,A}(f), y) \in \mathcal{P}_\omega(A)^X \times Y = H_{NAut(X,Y)}(A).$$

- For all  $f \in ((M \times A)^*)^X = H_{WAut^*(X,M,Y)}(A)$  and  $y \in Y$ ,

$$\tau_{5,A}(f, y) = (\tau_{3,A}(f), y) \in (M_\omega^A)^X \times Y = H_{WAut(X,M,Y)}(A).$$

- For all  $f \in ((\mathbb{R}_{\geq 0} \times A)^*)^X = H_{WAut^*(X,\mathbb{R}_{\geq 0},Y)}(A)$  and  $y \in Y$ ,

$$\tau_{6,A}(f, y) = (\lambda x. \lambda a. (\sum_{(a,r) \in f(x)} r) / \sum_{(b,r) \in f(x)} r, y) \in (M_{\{1\}}^A)^X \times Y = H_{PrAut(X,Y)}(A).$$

- For all  $(as_c)_{c:e \rightarrow s \in C} \in \prod_{c:e \rightarrow s \in C} A_e^* = H_{NTAcc^*(\Sigma)}(A)$ ,

$$\begin{aligned} \tau_{7,A}((as_c)_{c:e \rightarrow s \in C}) &= (\{\pi_i(as_c) \mid 1 \leq i \leq |as_c|\})_{c:e \rightarrow s \in C} \\ &\in \prod_{c:e \rightarrow s \in C} \mathcal{P}_\omega(A_e) = H_{NTAcc(\Sigma)}(A). \end{aligned}$$

$\tau_1, \dots, \tau_7$  are surjective natural transformations.

Hence by Lemma **WEAKFIN**, for all

$$\Sigma' \in \left\{ \begin{array}{l} WStream(X, M), NMed(X, M), WMed(X, M), NAut(X, Y), \\ WAut(X, M, Y), PrAut(X, Y), NTAcc(\Sigma) \end{array} \right\}$$

there is destructive polynomial signature  $\Sigma$  such that a final  $\Sigma'$ -algebra is given by a quotient of the final  $\Sigma$ -algebra.

In order to illustrate how final models of non-polynomial destructive signatures look like we take a closer look at  $\tau_2$  (see above):

\*\*\*\* Let  $\Sigma = (S, \mathcal{I}, F)$  be a destructive signature,

$$F' = \{f' : s \rightarrow (e^*)^X \mid f : s \rightarrow \mathcal{P}_\omega(e)^X \in F\}$$

and  $\Sigma' = (S, \mathcal{I}, F')$ .

A surjective natural transformation  $\tau : H_{\Sigma'} \rightarrow H_\Sigma$  is defined as follows:

For all  $S$ -sorted sets  $A$ ,  $b \in H_{\Sigma'}(A)$ ,  $f : s \rightarrow (e_1 + \dots + e_n)^X \in F$  and  $x \in X$ ,

$$\pi_f(\tau_A(b))(x) = \begin{cases} (\{a_1, \dots, a_k\}, i) & \text{if } e_i = \mathcal{P}_\omega(e) \text{ for some } e \\ & \text{and } \pi_f(b)(x) = ((a_1, \dots, a_k), i), \\ \pi_f(b)(x) & \text{otherwise.} \end{cases}$$

Since  $A = \nu\Sigma'$  is final in  $Alg_{\Sigma'}$ , Lemma **WEAKFIN** implies that  $A$  is weakly final in  $Alg_{\Sigma}$  if  $F$  is interpreted as follows: For all  $f : s \rightarrow (e_1 + \dots + e_n)^X$ ,

$$f^{\mathcal{A}} = \pi_f \circ \tau_{A,s} \circ \langle f'^{\mathcal{A}} \rangle_{f \in F},$$

i.e., for all  $a \in A_s$  and  $x \in X$ ,

$$f^{\mathcal{A}}(a)(x) = \begin{cases} (\{a_1, \dots, a_k\}, i) & \text{if } e_i = \mathcal{P}_{\omega}(e) \text{ for some } e \text{ and } f'^{\mathcal{A}}(a)(x) = ((a_1, \dots, a_k), i), \\ f'^{\mathcal{A}}(a)(x) & \text{otherwise.} \end{cases}$$

Moreover,

$$\nu\Sigma =_{def} A/\sim \text{ is final in } Alg_{\Sigma}$$

where  $\sim$  is the greatest  $\Sigma$ -congruence on  $A$ , i.e., the union of all  $S$ -sorted binary relations  $\sim$  on  $A$  such that for all  $s \in S$  and  $a, b \in A_s$ ,  $a \sim_s b$  implies  $f^{\mathcal{A}}(a) \sim_{(e_1 + \dots + e_n)^X} f^{\mathcal{A}}(b)$ , i.e., for all  $x \in X$ ,

$$\begin{aligned} \{a_1, \dots, a_k\} \sim_{e_i} \{b_1, \dots, b_l\} & \quad \text{if } e_i = \mathcal{P}_{\omega}(e) \text{ for some } e, f'^{\mathcal{A}}(a)(x) = ((a_1, \dots, a_k), i) \\ & \quad \text{and } f'^{\mathcal{A}}(b)(x) = ((b_1, \dots, b_l), i), \\ f'^{\mathcal{A}}(a)(x) \sim f'^{\mathcal{A}}(b)(x) & \quad \text{otherwise.} \end{aligned}$$

By Lambek's Lemma (2),  $\alpha : A \rightarrow H_\Sigma(A)$  as defined in diagram (1) of chapter 12 is iso and thus for all  $f : s \rightarrow (e_1 + \cdots + e_n)^X \in F$  and  $t \in \nu\Sigma_s$ ,

$$f^{\nu\Sigma}(t) = \pi_f(t) = t(f).$$

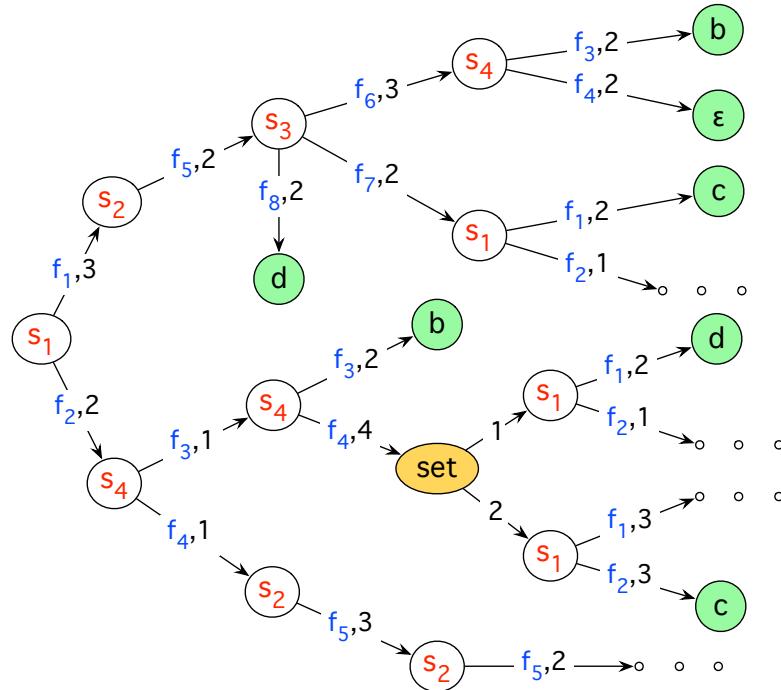
Hence for all  $\Sigma$ -algebras  $A$ ,  $a \in A_s$  and  $x \in X$ ,  $f^A(a)(x) = (b, i)$  implies

$$\begin{aligned} \text{unfold}^A(a)(f)(x) &= f^{\nu\Sigma}(\text{unfold}^A(a))(x) = \text{unfold}_{(e_1 + \cdots + e_n)^X}^A(f^A(a))(x) \\ &= \text{unfold}_{e_1 + \cdots + e_n}^A(f^A(a)(x)) = \text{unfold}_{e_1 + \cdots + e_n}^A(b, i) = (\text{unfold}_{e_i}^A(b), i). \end{aligned}$$

Moreover, for  $A = \nu\Sigma$  and all  $s \in S$ ,

$$\begin{aligned} A_s \cong H_\Sigma(A)_s &= \{t : F \rightarrow (A \times \mathbb{N})^X \mid \forall f : s \rightarrow (e_1 + \cdots + e_n)^X \in F \ \forall x \in X \\ &\quad \exists 1 \leq i \leq n : t(f)(x) \in A_{e_i} \times \{i\}\}. \end{aligned}$$

Hence  $\nu\Sigma$  can be represented as a quotient of the set  $\text{co}T_\Sigma$  (see section 19.13: Coterm functors).



A  $\Sigma$ -coterm. Each root of a subcoterm is labelled with its sort.

## Recursive functions

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature,  $V$  be a  $\mathcal{T}(S, \mathcal{I})$ -sorted set of variables and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .

### Theorem RECFUN1

Let  $C\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial subsignature of  $\Sigma$ ,  $\mathcal{A}|_{C\Sigma}$  be initial in  $Alg_{C\Sigma}$  and  $D = \{d_s : s \rightarrow e_s \mid s \in S\} \subseteq V$ . For all  $c : e \rightarrow s \in C$ , let  $\bar{c} : e[e_s/s \mid s \in S] \rightarrow e_s$  be a closed  $\lambda\Sigma$ -term over  $V$  such that  $var(\bar{c}) \cap D = \emptyset$ .

There is a unique solution  $g \in hval(V, A)$  of

$$\bigwedge_{c:e \rightarrow s \in C} d_s \circ c = \bar{c} \circ D_e \quad (1)$$

and, equivalently, of

$$\bigwedge_{s \in S} d_s = \text{case}\{c.\bar{c} \circ D_e\}_{c:e \rightarrow s \in C} \quad (2)$$

in  $\mathcal{A}$  (see  $\Sigma$ -formulas and Functors for constructive signatures). For the definition of the type instance  $D_e : e \rightarrow e[e_s/s \mid s \in S]$ , see chapter 8.

If  $e \in \mathcal{I}$ , then  $D_e = id_e$  and thus can be omitted in (1) and (2).

The set of factors of (1) is called an **inductive definition of  $D$  on  $\mathcal{A}$** .

*Proof.* Let  $\mathcal{B}$  be the  $C\Sigma$ -algebra that is defined as follows:

For all  $s \in S$ ,  $\mathcal{B}(s) = \mathcal{A}(e_s)$  and for all  $c : e \rightarrow s$ ,  $c^{\mathcal{B}} = \bar{c}^{\mathcal{A}}$ .

Since  $\mathcal{A}$  is initial in  $Alg_{\Sigma}$ ,  $fold^{\mathcal{B}} : \mathcal{A} \rightarrow \mathcal{B}$  is the unique  $Alg_{H_{\Sigma}}$ -morphism from  $\alpha = ([c^{\mathcal{A}}]_{c:e \rightarrow s \in C})_{s \in S}$  to  $\beta = ([c^{\mathcal{B}}]_{c:e \rightarrow s \in C})_{s \in S}$ , i.e.,  $fold^{\mathcal{B}}$  is the unique  $S$ -sorted function such that the following diagram commutes for all  $s \in S$ :

$$\begin{array}{ccc} H_{\Sigma}(A)_s & \xrightarrow{\alpha_s} & A_s \\ \downarrow & (3) & \downarrow \\ H_{\Sigma}(fold^{\mathcal{B}})_s & & \\ \downarrow & & \\ H_{\Sigma}(B)_s & \xrightarrow{\beta_s} & B_s \end{array}$$

By Lemma ISO (1), (3) commutes iff (4) commutes:

$$\begin{array}{ccc} H_{\Sigma}(A)_s & \xleftarrow{\alpha_s^{-1}} & A_s \\ \downarrow & (4) & \downarrow \\ H_{\Sigma}(fold^{\mathcal{B}})_s & & \\ \downarrow & & \\ H_{\Sigma}(B)_s & \xrightarrow{\beta_s} & B_s \end{array}$$

For all  $s \in S$ , define  $g(d_s) = \text{fold}_s^{\mathcal{B}}$ .

(4) commutes iff  $g$  solves (2) in  $\mathcal{A}$ :

If (4) commutes, then for all  $s \in S$ ,

$$\begin{aligned}
 g(d_s) &= \text{fold}_s^{\mathcal{B}} \stackrel{(4)}{=} \beta_s \circ H_{\Sigma}(\text{fold}^{\mathcal{B}})_s \circ \alpha^{-1} = [c^{\mathcal{B}}]_{c:e \rightarrow s \in C} \circ (\coprod_{c:e \rightarrow s \in C} \text{fold}_e^{\mathcal{B}}) \circ [c^{\mathcal{A}}]_{c:e \rightarrow s \in C}^{-1} \\
 &\stackrel{(19) \text{ in chapter 2}}{=} [c^{\mathcal{B}} \circ \text{fold}_e^{\mathcal{B}}]_{c:e \rightarrow s \in C} \circ [c^{\mathcal{A}}]_{c:e \rightarrow s \in C}^{-1} = [\bar{c}^{\mathcal{A}} \circ \text{fold}_e^{\mathcal{B}}]_{c:e \rightarrow s \in C} \circ [c^{\mathcal{A}}]_{c:e \rightarrow s \in C}^{-1} \\
 &= [g^*(\bar{c}) \circ g^*(D_e)]_{c:e \rightarrow s \in C} \circ [c^{\mathcal{A}}]_{c:e \rightarrow s \in C}^{-1} = [g^*(\bar{c} \circ D_e)]_{c:e \rightarrow s \in C} \circ [c^{\mathcal{A}}]_{c:e \rightarrow s \in C}^{-1} \\
 &= g^*(\text{case}\{c.\bar{c} \circ D_e\}_{c:e \rightarrow s \in C}),
 \end{aligned}$$

i.e.,  $g$  solves (2) in  $\mathcal{A}$ .

Conversely, if  $g$  solves (2) in  $\mathcal{A}$ , then for all  $s \in S$ ,

$$\begin{aligned}
 \text{fold}_s^{\mathcal{B}} &= g(d_s) \stackrel{(2)}{=} g^*(\text{case}\{c.\bar{c} \circ D_e\}_{c:e \rightarrow s \in C}) = \dots \text{ (see above) } \dots \\
 &= \beta_s \circ H_{\Sigma}(\text{fold}^{\mathcal{B}})_s \circ \alpha^{-1},
 \end{aligned}$$

i.e., (4) commutes.



Roughly said, an inductive definition specifies destructors in terms of constructors—exactly one destructor for each sort. As we have seen above, this is not a restriction: several destructors for the same sort can always be combined into a single one by building the [product of their targets](#).

If the product has several factors, this means that the inductive definition defines several functions (with the same domain) simultaneously, possibly in a mutually recursive way.

Some of them may serve only as auxiliary functions like the identity function that is needed if certain arguments of the function  $f$  to be defined occur outside recursive calls. Then  $f$  is called a **paramorphism** and the defining equations follow the pattern of primitive recursion (see chapter 12). In turn, paramorphisms are adjoint folds for an adjunction of the form  $(\Delta^I : \mathbf{Set} \rightarrow \mathbf{Set}^I, \prod_{i \in I} : \mathbf{Set}^I \rightarrow \mathbf{Set})$  (see chapter 24).

The proof of Theorem RECFUN1 also reveals that (1) is equivalent to the compatibility of  $d$  with the  $\Sigma$ -homomorphism  $\text{fold}^{\mathcal{B}} : \mathcal{A} \rightarrow \mathcal{B}$ . Hence, basically, the unique solvability of (1) follows from the uniqueness of a  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $\mathcal{B}$ , which in turn follows from the initiality of  $\mathcal{A}$  in  $\text{Alg}_{\Sigma}$ .

## Theorem RECFUN2

Let  $D\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial subsignature of  $\Sigma$ ,  $\mathcal{A}|_{D\Sigma}$  be final in  $Alg_{D\Sigma}$  and  $C = \{c_s : e_s \rightarrow s \mid s \in S\} \subseteq V$ . For all  $d : s \rightarrow e \in D$ , let  $\bar{d} : e_s \rightarrow e[e_s/s \mid s \in S]$  be a closed  $\lambda\Sigma$ -term over  $V$  such that  $var(\bar{d}) \cap C = \emptyset$ .

There is a unique solution  $g \in hval(V, A)$  of

$$\bigwedge_{d:s \rightarrow e \in D} d \circ c_s = C_e \circ \bar{d} \quad (1)$$

and, equivalently, of

$$\bigwedge_{s \in S} c_s = obj\{d.C_e \circ \bar{d}\}_{d:s \rightarrow e' \in D} \quad (2)$$

in  $\mathcal{A}$  (see  $\Sigma$ -formulas and [Functors for destructive signatures](#)). For the definition of the type instance  $C_e : e[e_s/s \mid s \in S] \rightarrow e$ , see chapter 8.

If  $e \in \mathcal{I}$ , then  $C_e = id_e$  and thus can be omitted in (1) and (2).

The set of factors of (1) is called a **coinductive definition of  $C$  on  $\mathcal{A}$** .

*Proof.* Let  $\mathcal{B}$  be the  $D\Sigma$ -algebra that is defined as follows:

For all  $s \in S$ ,  $\mathcal{B}(s) = \mathcal{A}(e_s)$  and for all  $d : s \rightarrow e$ ,  $d^{\mathcal{B}} = \bar{d}^{\mathcal{A}}$ .

Since  $\mathcal{A}$  is final in  $Alg_{\Sigma}$ ,  $unfold^{\mathcal{B}} : \mathcal{B} \rightarrow \mathcal{A}$  is the unique  $Alg_{H_{\Sigma}}$ -morphism from  $\beta = (\langle d^{\mathcal{B}} \rangle_{d:s \rightarrow e \in D})_{s \in S}$  to  $\alpha = (\langle d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D})_{s \in S}$ , i.e.,  $unfold^{\mathcal{B}}$  is the unique  $S$ -sorted function such that the following diagram commutes for all  $s \in S$ :

$$\begin{array}{ccc}
 H_{\Sigma}(A)_s & \xleftarrow{\alpha} & A_s \\
 \uparrow & & \uparrow \\
 H_{\Sigma}(unfold^{\mathcal{B}})_s & & (3) \\
 \downarrow & & \downarrow unfold_s^{\mathcal{B}} \\
 H_{\Sigma}(B)_s & \xleftarrow{\beta} & B_s
 \end{array}$$

By Lemma ISO (2), (3) commutes iff (4) commutes:

$$\begin{array}{ccc}
 H_{\Sigma}(A)_s & \xrightarrow{\alpha^{-1}} & A_s \\
 \uparrow & & \uparrow \\
 H_{\Sigma}(unfold^{\mathcal{B}})_s & & (4) \\
 \downarrow & & \downarrow unfold_s^{\mathcal{B}} \\
 H_{\Sigma}(B)_s & \xleftarrow{\beta} & B_s
 \end{array}$$

For all  $s \in S$ , define  $g(c_s) = unfold^{\mathcal{B}}$ .

(4) commutes iff  $g$  solves (2) in  $\mathcal{A}$ :

If (4) commutes, then for all  $s \in S$ ,

$$\begin{aligned}
 g(c_s) &= \text{unfold}_s^{\mathcal{B}} \stackrel{(4)}{=} \alpha^{-1} \circ H_{\Sigma}(\text{unfold}^{\mathcal{B}})_s \circ \beta_s \\
 &= \langle d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}^{-1} \circ (\prod_{d:s \rightarrow e \in D} \text{unfold}_e^{\mathcal{B}}) \circ \langle d^{\mathcal{B}} \rangle_{d:s \rightarrow e \in D} \\
 &\stackrel{(8) \text{ in chapter 2}}{=} \langle d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}^{-1} \circ \langle \text{unfold}_e^{\mathcal{B}} \circ d^{\mathcal{B}} \rangle_{d:s \rightarrow e \in D} \\
 &= \langle d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}^{-1} \circ \langle \text{unfold}_e^{\mathcal{B}} \circ \bar{d}^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D} = \langle d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}^{-1} \circ \langle g^*(C_e) \circ g^*(\bar{d}) \rangle_{d:s \rightarrow e \in D} \\
 &= \langle d^{\mathcal{A}} \rangle_{d:s \rightarrow e \in D}^{-1} \circ \langle g^*(C_e \circ \bar{d}) \rangle_{d:s \rightarrow e \in D} = g^*(\text{obj}\{d.C_e \circ \bar{d}\}_{d:s \rightarrow e \in D}),
 \end{aligned}$$

i.e.,  $g$  solves (2) in  $\mathcal{A}$ .

Conversely, if  $g$  solves (2) in  $\mathcal{A}$ , then for all  $s \in S$ ,

$$\begin{aligned}
 \text{unfold}_s^{\mathcal{B}} &= g(c_s) \stackrel{(2)}{=} g^*(\text{obj}\{d.C_e \circ \bar{d}\}_{d:s \rightarrow e' \in D}) = \dots \text{ (see above) } \dots \\
 &= \alpha^{-1} \circ H_{\Sigma}(\text{unfold}^{\mathcal{B}})_s \circ \beta,
 \end{aligned}$$

i.e., (4) commutes. □

Roughly said, a coinductive definition specifies constructors in terms of destructors—exactly one destructor for each sort. As we have seen above, this is not a restriction: several constructors for the same sort can always be combined into a single one by building the [sum of their domains](#).

If the sum has several summands, this means that the coinductive definition defines several functions (with the same range) simultaneously, possibly in a mutually recursive way. Some of them may serve only as auxiliary functions like the identity function that is needed if certain arguments of the function  $f$  to be defined occur outside recursive calls of  $f$ . Then  $f$  is called an **apomorphism** and the defining equations follow the pattern of primitive corecursion (see chapter 12). In turn, apomorphisms are adjoint unfolds for an adjunction of the form  $(\coprod_{i \in I} : \mathbf{Set}^I \rightarrow \mathbf{Set}, \Delta^I : \mathbf{Set} \rightarrow \mathbf{Set}^I)$  (see chapter 24).

The proof of Theorem RECFUN2 also reveals that (1) is equivalent to the compatibility of  $c$  with the  $\Sigma$ -homomorphism  $\text{unfold}^{\mathcal{B}} : \mathcal{B} \rightarrow \mathcal{A}$ . Hence, basically, the unique solvability of (1) follows from the uniqueness of a  $\Sigma$ -homomorphism from  $\mathcal{B}$  to  $\mathcal{A}$ , which in turn follows from the finality of  $\mathcal{A}$  in  $\text{Alg}_{\Sigma}$ .

To coinductive definitions of Theorem RECFUN2 restrict the “recursive calls” of the functions to be defined to outermost positions. A definitional schema that admits the proper embedding of recursive calls requires a different proof.

Moreover, functions to be defined by mutual recursion need no longer be turned into their sum extension (see examples *blink*"", *eosum* and *exchange*" given below).

On the one hand, the new schema is more general than (1). On the other hand, it restricts the structure of terms on the right-hand sides of defining equations insofar as destructors may occur only at innermost term positions.

## Theorem RECFUN3

Let  $D\Sigma = (S, \mathcal{I}, D)$  and  $C\Sigma = (S, \mathcal{I}, C)$  be a destructive resp. constructive polynomial subsignature of  $\Sigma$ ,  $\mathcal{A}|_{D\Sigma}$  be final in  $Alg_{D\Sigma}$  and for all  $s \in S$ ,  $c : e \rightarrow s \in C$  and  $d : s \rightarrow e'$ , let  $e_s = \prod_{d:s \rightarrow e' \in D} e'$ ,  $T = \{\langle d \rangle_{d:s \rightarrow e' \in D} : s \rightarrow e_s \mid s \in S\}$  and  $\bar{c}_d : e[e_s/s \mid s \in S] \rightarrow e'$  be a closed  $\lambda$ - $C\Sigma$ -term over  $V$  such that  $free(\bar{c}_d) \subseteq C \subseteq V$ .

There is a unique solution  $g : V \rightarrow A$  of

$$\bigwedge_{c:e \rightarrow s \in C, d:s \rightarrow e' \in D} d \circ c = \bar{c}_d \circ T_e \quad (1)$$

and, equivalently, of

$$\bigwedge_{c:e \rightarrow s \in C} c = obj\{d.\bar{c}_d \circ T_e\}_{d:s \rightarrow e' \in D} \quad (2)$$

in  $\mathcal{A}$ . For the definition of the type instance  $T_e : e \rightarrow e[e_s/s \mid s \in S]$ , see chapter 8.

Theorem RECFUN3 justifies it to call the set of factors of (1) a **biinductive definition of  $C$  on  $\mathcal{A}$** .

*Proof.* Let  $C\Sigma(A) = (S, \mathcal{I}(A), C(A))$  be the grounding of  $C\Sigma$  on  $A$  (see **Term folding**) and  $\mathcal{B}$  be the initial  $C\Sigma(A)$ -algebra with carrier  $T_{C\Sigma(A)}$  and the following interpretation of  $D$ :

For all  $c : e \rightarrow s \in C$ ,  $d : s \rightarrow e' \in D$ ,  $u \in T_{C\Sigma(A), e}$  and  $a \in A_s$ ,

$$d^{\mathcal{B}}(c(u)) =_{\text{def}} \bar{c}_d^{\mathcal{B}}(T_e^{\mathcal{B}}(u)) \in T_{C\Sigma(A), e'}, \quad (3)$$

$$d^{\mathcal{B}}(\text{val}_s(a)) =_{\text{def}} \text{val}_{e'}^{\mathcal{B}}(d^{\mathcal{A}}(a)) \in T_{C\Sigma(A), e'}, \quad (4)$$

where  $\text{val}_{e'}^{\mathcal{B}} : A_{e'} \rightarrow T_{C\Sigma(A), e'}$  is the  $e'$ -component of the lifting to  $\mathcal{T}_q(S, \mathcal{I})$  of the  $S$ -sorted function  $\text{val}^{\mathcal{B}} : A \rightarrow T_{C\Sigma(A)}$ , which maps  $a \in A_s$  to the term  $\text{val}_s(a)$  (see  **$\Sigma$ -terms and -coterms**).

Since  $\mathcal{A}|_{D\Sigma}$  is final in  $\text{Alg}_{D\Sigma}$ , there is a unique  $D\Sigma$ -homomorphism  $\text{unfold}^{\mathcal{B}} : \mathcal{B} \rightarrow \mathcal{A}$ .

By (4),  $\text{val}^{\mathcal{B}}$  and thus  $\text{unfold}^{\mathcal{B}} \circ \text{val}^{\mathcal{B}} : A \rightarrow A$  are  $D\Sigma$ -homomorphic. Hence

$$\text{unfold}^{\mathcal{B}} \circ \text{val}^{\mathcal{B}} = id_A, \quad (5)$$

again because  $\mathcal{A}|_{D\Sigma}$  is final in  $\text{Alg}_{D\Sigma}$ .

$\mathcal{A}$  is a  $C\Sigma(A)$ -algebra: For all  $c : e \rightarrow s \in C$  and  $s \in S$ ,

$$c^{\mathcal{A}} =_{def} unfold_s^{\mathcal{B}} \circ c^{\mathcal{B}} \circ val_e^{\mathcal{B}}, \quad (6)$$

$$val_s^{\mathcal{A}} =_{def} id_{A_s}. \quad (7)$$

Hence for all  $d : s \rightarrow e'$  and  $a \in A_s$ ,

$$\begin{aligned} d^{\mathcal{A}}(c^{\mathcal{A}}(a)) &\stackrel{(6)}{=} d^{\mathcal{A}}(unfold_s^{\mathcal{B}}(c^{\mathcal{B}}(val_e^{\mathcal{B}}(a)))) = d^{\mathcal{A}}(unfold_s^{\mathcal{B}}(c(val_e^{\mathcal{B}}(a)))) \\ &\stackrel{unfold^{\mathcal{B}}}{=} unfold_e^{\mathcal{B}}(d^{\mathcal{B}}(c(val_e^{\mathcal{B}}(a)))) \stackrel{(3)}{=} unfold_e^{\mathcal{B}}(\bar{c}_d^{\mathcal{B}}(T_e^{\mathcal{B}}(val_e^{\mathcal{B}}(a)))) \\ &\stackrel{unfold^{\mathcal{B}} \text{ } C\Sigma\text{-hom.}, \text{ Lemma HOMARR}}{=} \bar{c}_d^{\mathcal{A}}(unfold_{e[e_s/s|s \in S]}^{\mathcal{B}}(T_e^{\mathcal{B}}(val_e^{\mathcal{B}}(a)))) \\ &\stackrel{unfold^{\mathcal{B}}}{=} \bar{c}_d^{\mathcal{A}}(T_e^{\mathcal{A}}(unfold_e^{\mathcal{B}}(val_e^{\mathcal{B}}(a)))) \stackrel{(5)}{=} \bar{c}_d^{\mathcal{A}}(T_e^{\mathcal{A}}(a)), \end{aligned}$$

i.e.,  $\mathcal{A}$  satisfies (1), in other words:  $g : V \rightarrow A$  with  $g(c) = unfold_s^{\mathcal{B}} \circ c^{\mathcal{B}} \circ val_e^{\mathcal{B}}$  for all  $c \in C$  solves (1) in  $\mathcal{A}$ . (8)

Further consequences:

(9) The greatest  $D\Sigma$ -congruence  $\sim$  on  $\mathcal{B}$  is a  $C\Sigma$ -congruence: By (3),  $\mathcal{B}$  satisfies (2). Hence by Lemma MOD, the  $C\Sigma$ -congruence closure  $\sim_C$  of  $\sim$  is a  $D\Sigma$ -congruence. Since  $\sim$  is both the greatest  $D\Sigma$ -congruence and a subrelation of  $\sim_C$ ,  $\sim$  is equal to  $\sim_C$  and thus a  $C\Sigma$ -congruence.

By Lemma FIN (4),  $\ker(\text{unfold}^{\mathcal{B}})$  is the greatest  $D\Sigma$ -congruence on  $\mathcal{B}$  (see chapter 15). Hence by (9),  $\ker(\text{unfold}^{\mathcal{B}})$  is a  $C\Sigma$ -congruence, i.e., for all  $c : e \rightarrow s \in C$  and  $t, t' \in T_{C\Sigma}(A)_e$ ,  $\text{unfold}^{\mathcal{B}}(t) = \text{unfold}^{\mathcal{B}}(t')$  implies  $\text{unfold}^{\mathcal{B}}(c^{\mathcal{B}}(t)) = \text{unfold}^{\mathcal{B}}(c^{\mathcal{B}}(t'))$ . Consequently and since by (5),  $\text{unfold}^{\mathcal{B}} : B \rightarrow A$  is surjective,  $A$  is the carrier of the  $C\Sigma$ -algebra  $\mathcal{A}'$ , which interprets  $C$  as follows:

For all  $c : e \rightarrow s \in C$  and  $t, t' \in T_{C\Sigma}(A)_e$ ,

$$c^{\mathcal{A}'}(\text{unfold}^{\mathcal{B}}(t)) =_{\text{def}} \text{unfold}^{\mathcal{B}}(c^{\mathcal{B}}(t)). \quad (10)$$

(11)  $\mathcal{A}$  and  $\mathcal{A}'$  interpret  $C$  in the same way: For all  $c : e \rightarrow s \in C$  and  $a \in A_e$ ,

$$c^{\mathcal{A}'}(a) \stackrel{(5)}{=} c^{\mathcal{A}'}(\text{unfold}^{\mathcal{B}}(\text{val}^{\mathcal{B}}(a))) \stackrel{(9)}{=} \text{unfold}^{\mathcal{B}}(c^{\mathcal{B}}(\text{val}^{\mathcal{B}}(a))) \stackrel{(6)}{=} c^{\mathcal{A}}(a).$$

(12)  $\text{unfold}^{\mathcal{B}}$  is  $C\Sigma(A)$ -homomorphic: For all  $c : e \rightarrow s \in C$  and  $t \in T_{C\Sigma}(A)_e$ ,

$$\text{unfold}^{\mathcal{B}}(c^{\mathcal{B}}(t)) \stackrel{(10)}{=} c^{\mathcal{A}'}(\text{unfold}^{\mathcal{B}}(t)) \stackrel{(11)}{=} c^{\mathcal{A}}(\text{unfold}^{\mathcal{B}}(t)).$$

For all  $s \in S$  and  $a \in A_s$ ,

$$\text{unfold}^{\mathcal{B}}(\text{val}_s^{\mathcal{B}}(a)) \stackrel{(5)}{=} a \stackrel{(7)}{=} \text{val}_s^{\mathcal{A}}(a) \stackrel{\text{unfold}^{\mathcal{B}}}{{}_{D\Sigma-\text{hom.}}} \text{val}_s^{\mathcal{A}}(\text{unfold}_{A_s}^{\mathcal{B}}(a)).$$

(13)  $fold^{\mathcal{A}} = unfold^{\mathcal{B}}$ :  $fold^{\mathcal{A}}$  is the unique  $C\Sigma(A)$ -homomorphism from the initial  $C\Sigma(A)$ -algebra  $T_{C\Sigma}(A)$  to  $\mathcal{A}$ , which by (6) and (7) is also a  $C\Sigma(A)$ -algebra. Hence by (12),  $fold^{\mathcal{A}}$  coincides with  $unfold^{\mathcal{B}}$ .

It remains to show that  $g$  in (8) is the only solution of (1) in  $\mathcal{A}$ .

Let  $h : V \rightarrow A$  be an arbitrary solution of (1) in  $\mathcal{A}$  and  $\sim$  be the least  $S$ -sorted equivalence relation on  $A$  such that  $\sim$  contains  $\Delta_A$  and for all  $c : e \rightarrow s \in C$  and  $a, b \in A_e$ ,  $a \sim b$  implies  $g(c)(a) \sim h(c)(b)$ .

Suppose that  $\sim$  is a  $D\Sigma$ -congruence.

For all  $c : e \rightarrow s \in C$  and  $a \in A_e$ ,  $a \sim a$  implies  $g(c)(a) \sim h(c)(a)$  and thus  $g(c)(a) = h(c)(a)$  because by Lemma FIN (3),  $\Delta_A$  is the only  $D\Sigma$ -congruence on  $A$ . Hence  $g|_C = h|_C$ .

It remains to show (by induction on the definition of  $\sim$ ) that  $\sim$  is a  $D\Sigma$ -congruence.

Let  $s \in S$ ,  $a \sim_s b$  and  $d : s \rightarrow e' \in D$ .

*Case 1:*  $a = b$ . Hence  $d^{\mathcal{A}}(a) = d^{\mathcal{A}}(b)$  and thus  $d^{\mathcal{A}}(a) \sim d^{\mathcal{A}}(b)$  because  $\sim$  is reflexive.

*Case 2:*  $b \sim a$ . By induction hypothesis,  $d^{\mathcal{A}}(b) \sim d^{\mathcal{A}}(a)$  and thus  $d^{\mathcal{A}}(a) \sim d^{\mathcal{A}}(b)$  because  $\sim$  is symmetric.

*Case 3:*  $a \sim a'$  and  $a' \sim b$  for some  $a' \in A_s$ . By induction hypothesis,  $d^{\mathcal{A}}(a) \sim d^{\mathcal{A}}(a')$  and  $d^{\mathcal{A}}(a') \sim d^{\mathcal{A}}(b)$ . Hence  $d^{\mathcal{A}}(a) \sim d^{\mathcal{A}}(b)$  because  $\sim$  is transitive.

*Case 4:* There are  $c : e \rightarrow s \in C$ ,  $a', b' \in A_e$  such that  $a = g(c)(a')$ ,  $b = h(c)(b')$  and  $a' \sim_e b'$ . By induction hypothesis,  $T_e^{\mathcal{A}}(a') \sim_{e[e_s/s|s \in S]} T_e^{\mathcal{A}}(b')$ . Hence

$$\begin{aligned} d^{\mathcal{A}}(a) &= d^{\mathcal{A}}(g(c)(a')) \stackrel{(1)}{=} \bar{c}_d^{\mathcal{A}}(T_e^{\mathcal{A}}(a')) \xrightarrow{\text{Lemma CONGCL}} \bar{c}_d^{\mathcal{A}}(T_e^{\mathcal{A}}(b')) \stackrel{(1)}{=} d^{\mathcal{A}}(h(c)(b')) \\ &= d^{\mathcal{A}}(b). \end{aligned}$$
□

Crucial arguments in the proof of Theorem RECFUN3 originate from the proof of [146], Theorem 3.1, which shows that certain stream equations have unique solutions in final stream algebras and thus justifies their designation as *definitions* of stream constructors (see also [71], Appendices A.5 and A.6; [88], section 4; [64], section 8.2).

In a rather informal way, [148], Thm. A.1, provides a schema for biinductive definitions of stream constructors, called **differential equations**. The proof sketch of this theorem has been carried out in more detail in [48].

Similar schemas for biinductively defined functions dealing with infinite binary trees over a semiring, nondeterministic transition systems, Mealy automata, concurrent processes and formal power series are given in [155], section 3, [34], [65], [74, 142, 77] and [146], section 9; respectively.

Theorem RECFUN3 provides the coincidence of a *fold* and an *unfold* (see (13) in the proof), which is often regarded as a correspondence between **denotational semantics** and **operational semantics** (e.g., in [74]).

Consequently, biinductive definitions are closely related to **structural-operational (SOS) rules**, which, e.g., determine the syntax (constructors) and operational semantics of programming languages or process calculi. Here the destructors often come as multivalued transition functions. Hence SOS rules are biinductive definitions in relational form.

For instance, the following SOS rules reproduce the biinductive definition of regular operators (see below):

$$\frac{t \xrightarrow{\delta} t', u \xrightarrow{\delta} u'}{par(t, u) \xrightarrow{\delta} \lambda x. par(t'(x), u'(x))}$$

$$\frac{}{t \xrightarrow{\delta} t', u \xrightarrow{\delta} u'}$$

$$seq(t, u) \xrightarrow{\delta} \lambda x. if \beta(t) then par(seq(t'(x), u), u'(x)) else seq(t'(x), u)$$

$$\frac{t \xrightarrow{\delta} t'}{iter(t) \xrightarrow{\delta} \lambda x. seq(t'(x), iter(t))}$$

$$\overline{eps \xrightarrow{\delta} \lambda x. base(\emptyset)}$$

$$\overline{base(B) \xrightarrow{\delta} \lambda x. if \ x \in B \ then \ eps \ else \ base(\emptyset)}$$

$$\frac{t \xrightarrow{\beta} m, u \xrightarrow{\beta} n}{par(t, u) \xrightarrow{\beta} max(m, n)}$$

$$\frac{t \xrightarrow{\beta} m, u \xrightarrow{\beta} n}{seq(t, u) \xrightarrow{\beta} m * n}$$

$$\overline{iter(t) \xrightarrow{\beta} 0}$$

$$\overline{eps \xrightarrow{\beta} 1}$$

$$\overline{base(B) \xrightarrow{\beta} 0}$$

Biinductive definitions have also been represented in a more abstract form (see, e.g., [26, 37, 70, 77, 86, 164], which does not employ the term algebra  $\mathcal{B}$  of Theorem RECFUN3. Instead, this representation involves a distributive law  $\lambda$  (see chapter 22). Adapting a non-trivial set of defining equations to this framework might become difficult—as the simple example given in chapter 24 suggests.

In this context, some terms may differ from ours. For instance, [26] would call coinductive definitions *coiterative* and biinductive ones  $\lambda$ -*coiterative*.

## Bisimulation modulo constructors

Let  $D\Sigma = (S, \mathcal{I}, D)$  be a destructive signature,  $C\Sigma = (S, \mathcal{I}, C)$  be a constructive signature,  $\mathcal{A}$  be a  $C\Sigma \cup D\Sigma$ -algebra with carrier  $A$  and  $\sim$  be an  $S$ -sorted binary relation on  $A$ .

The least  $C\Sigma$ -congruence on  $\mathcal{A}$  that contains  $\sim$  is called the  **$C\Sigma$ -congruence closure of  $\sim$**  and denoted by  $\sim_C$ .

$\sim$  is a  **$\Sigma$ -bisimulation modulo  $C$  on  $\mathcal{A}$**  if for all  $d : s \rightarrow e \in D$  and  $a, b \in A_s$ ,

$$a \sim_s b \Rightarrow d^{\mathcal{A}}(a) \sim_{C,e} d^{\mathcal{A}}(b).$$

## Lemma MOD

Let the assumptions of Theorem RECFUN3 hold true,

$$\bigwedge_{c:e \rightarrow s \in C, d:s \rightarrow e' \in D} d \circ c = \bar{c}_d \circ T_e \quad (1)$$

be a biinductive definition of  $C$  on  $\mathcal{A}$  and  $\sim$  be a  $D\Sigma$ -bisimulation modulo  $C$  on  $\mathcal{A}$ .

Then  $\sim_C$  is a  $D\Sigma$ -congruence.

*Proof by induction on the definition of  $\sim_C$ .*

Let  $s \in S$ ,  $a \sim_{C,s} b$  and  $d : s \rightarrow e' \in D$ .

*Case 1:*  $a \sim b$ . Since  $\sim$  is a  $D\Sigma$ -bisimulation modulo  $C$ ,  $d^{\mathcal{A}}(a) \sim_C d^{\mathcal{A}}(b)$ .

*Case 2:*  $a = b$ . Hence  $d^{\mathcal{A}}(a) = d^{\mathcal{A}}(b)$  and thus  $d^{\mathcal{A}}(a) \sim_C d^{\mathcal{A}}(b)$  because  $\sim_C$  is reflexive.

*Case 3:*  $b \sim_C a$ . By induction hypothesis,  $d^{\mathcal{A}}(b) \sim_C d^{\mathcal{A}}(a)$  and thus  $d^{\mathcal{A}}(a) \sim_C d^{\mathcal{A}}(b)$  because  $\sim_C$  is symmetric.

Case 4:  $a \sim_C a'$  and  $a' \sim_C b$  for some  $a' \in A_s$ . By induction hypothesis,  $d^{\mathcal{A}}(a) \sim_C d^{\mathcal{A}}(a')$  and  $d^{\mathcal{A}}(a') \sim_C d^{\mathcal{A}}(b)$ . Hence  $d^{\mathcal{A}}(a) \sim_C d^{\mathcal{A}}(b)$  because  $\sim_C$  is transitive.

Case 5: There are  $c : e \rightarrow s \in C$ ,  $a', b' \in A_e$  such that  $a = c^{\mathcal{A}}(a')$ ,  $b = c^{\mathcal{A}}(b')$  and  $a' \sim_{C,e} b'$ . By induction hypothesis,  $T_e^{\mathcal{A}}(a') \sim_{C,e[e_s/s|s \in S]} T_e^{\mathcal{A}}(b')$ . Hence

$$d^{\mathcal{A}}(a) = d^{\mathcal{A}}(c^{\mathcal{A}}(a')) \stackrel{(1)}{=} \bar{c}_d^{\mathcal{A}}(T_e^{\mathcal{A}}(a')) \xrightarrow[\text{Lemma CONGCL}]{\sim_{C,e'}} \bar{c}_d^{\mathcal{A}}(T_e^{\mathcal{A}}(b')) \stackrel{(1)}{=} d^{\mathcal{A}}(c^{\mathcal{A}}(b')) = d^{\mathcal{A}}(b). \quad \square$$

## Sample inductively defined functions

Let  $C\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial subsignature of  $\Sigma$  and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  such that  $\mathcal{A}|_{C\Sigma}$  is initial in  $Alg_{C\Sigma}$ . Theorem RECFUN1 tells us that a set of inductively defined functions can be regarded as the unique solution

$$\{\textcolor{red}{sol}(d_s) : A_s \rightarrow A_{e_s} \mid s \in S\}$$

of the formula

$$\bigwedge_{c:e \rightarrow s \in C} \textcolor{red}{d}_s \circ c = \bar{c} \circ D_e \tag{1}$$

in the set  $D = \{d_s : s \rightarrow e_s \mid s \in S\}$  of  $\lambda\Sigma$ -term variables.

In each of the following examples, we start out from a given set  $E$  of equations whose conjunction is equivalent to (1), i.e., (1) is solvable in  $D$  iff  $E$  is solvable in the variables of  $E$ .

Often the solution of (1) is a product extension of the solution of  $E$  and a further function  $f$ . For instance, if  $f$  is an identity, then  $E$  specifies a primitive recursive function or paramorphism (see chapter 12).

If the equations for  $D$  are given in some applicative form, “recursive calls” of  $D$  may scatter around on the equations’ right-hand sides. In order to obtain the right-hand side of (1), they must be bundled into a single term  $D_e : e \rightarrow e[e_s/s \mid s \in S]$ . In turn, this requires a general definition of  $\bar{c}$ , not only for the  $D$ -images where  $\bar{c}$  is applied to in the equations.

In [75], the formation of a definition of  $\bar{c}$  is called a *generalization* step and carried out explicitly in derivations of inductive definitions. In the examples given below, generalizations arise implicitly as soon as  $\bar{c}$  is presented as a  $\lambda$ -term.

## Inductively defined functions on natural numbers

Let  $C\Sigma = Nat$  (see chapter 8). The corresponding instance of (the single factor of) (1) and thus the schema for interpreting  $d : nat \rightarrow e$  as an inductively defined function on  $\mathcal{A}|_{Nat}$  reads as follows:

$$d \circ zero = \overline{zero}, \quad (2)$$

$$d \circ succ = \overline{succ} \circ d. \quad (3)$$

Let  $z : 1, b : 2, k, m, n : nat, f : nat \rightarrow nat \in V, one =_{def} succ \circ zero$  and  $(=), (\leq) : nat \times nat \rightarrow 2$  be interpreted in  $\mathcal{A}|_{Nat}$  as usually.

1. Let  $d = (= zero) : nat \rightarrow 2$  be a variable for the test on zero with the equations

$$d(zero(z)) = 1,$$

$$d(succ(n)) = 0.$$

An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$d \circ zero = \lambda z.1,$$

$$d \circ succ = (\lambda b.0) \circ d.$$

2. Let  $\text{pred} : \text{nat} \rightarrow \text{nat}$  be a variable for the predecessor function on  $A_{\text{Nat}}$ . The equations

$$\begin{aligned}\text{pred}(\text{zero}(z)) &= \text{zero}, \\ \text{pred}(\text{succ}(n)) &= n\end{aligned}$$

for  $\text{pred}$  entail the equations

$$\begin{aligned}d(\text{zero}(z)) &= (\text{zero}, \text{zero}), \\ d(\text{succ}(k)) &= (\lambda(m, n).(n, \text{succ}(n)))(d(k))\end{aligned}$$

for  $d =^{\mathcal{A}} \langle \text{pred}, \text{id}_{\text{nat}} \rangle : \text{nat} \rightarrow \text{nat} \times \text{nat}$  (see  $\Sigma$ -formulas).

An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned}\textcolor{red}{d} \circ \text{zero} &= \langle \text{zero}, \text{zero} \rangle, \\ \textcolor{red}{d} \circ \text{succ} &= (\lambda(m, n).(n, \text{succ}(n))) \circ \textcolor{red}{d}.\end{aligned}$$

Moreover,  $\text{sol}(\text{pred}) = \pi_1 \circ \text{sol}(d)$ . Hence  $\text{pred}$  is a paramorphism (see chapter 12).

3. Let  $\text{add} : \text{nat} \times \text{nat} \rightarrow \text{nat}$  be a variable for addition on  $A_{\text{Nat}}$ . The equations

$$\begin{aligned}\text{add}(\text{zero}(z), n) &= n, \\ \text{add}(\text{succ}(m), n) &= \text{succ}(\text{add}(m, n))\end{aligned}$$

for  $add$  entail the equations

$$\begin{aligned} d(zero(z)) &= id_{nat}, \\ d(succ(m)) &= \lambda n. succ(d(m)(n)) = (\lambda f. \lambda n. succ(f(n)))(d(m)) \end{aligned}$$

for  $d =^A curry(add) : nat \rightarrow (nat \rightarrow nat)$ . An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} \textcolor{red}{d} \circ zero &= \lambda z. id_{nat}, \\ \textcolor{red}{d} \circ succ &= (\lambda m. \lambda f. \lambda n. succ(f(n))) \circ \textcolor{red}{d}. \end{aligned}$$

Moreover,  $sol(add) = uncurry(sol(d))$ .

4. ([167], Example 12 - a *histomorphism*; [69], Example 21) Let  $fib : nat \rightarrow nat$  be a variable for the Fibonacci function. The equations

$$\begin{aligned} fib(zero(z)) &= zero, \\ fib(succ(zero(z))) &= one, \\ fib(succ(succ(n))) &= add(fib(n), fib(succ(n))) \end{aligned}$$

for  $fib$  entail the equations

$$\begin{aligned} d(zero(z)) &= (zero, one), \\ d(succ(k)) &= (\lambda(m, n). (n, add(m, n)))(d(k)) \end{aligned}$$

for  $d =^A \langle fib, fib \circ succ \rangle : nat \rightarrow nat \times nat$ .

An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned}\textcolor{red}{d} \circ \text{zero} &= \langle \text{zero}, \text{one} \rangle, \\ \textcolor{red}{d} \circ \text{succ} &= (\lambda(m, n). (n, \text{add}(m, n))) \circ \textcolor{red}{d}.\end{aligned}$$

Moreover,  $\text{sol}(\text{fib}) = \pi_1 \circ \text{sol}(d)$ .

5. Let  $\text{mul} : \text{nat} \times \text{nat} \rightarrow \text{nat}$  be a variable for multiplication on  $A_{\text{Nat}}$ . The equations

$$\begin{aligned}\text{mul}(\text{zero}(z), n) &= \text{zero}, \\ \text{mul}(\text{succ}(m), n) &= \text{add}(\text{mul}(m, n), n)\end{aligned}$$

for  $\text{mult}$  entail the equations

$$\begin{aligned}d(\text{zero}(z)) &= \lambda n. \text{zero}, \\ d(\text{succ}(m)) &= \lambda n. \text{add}(d(m)(n), n) = (\lambda f. \lambda n. \text{add}(f(n), n))(d(m))\end{aligned}$$

for  $\textcolor{red}{d} =^{\mathcal{A}} \text{curry}(\text{mul}) : \text{nat} \rightarrow (\text{nat} \rightarrow \text{nat})$ . An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned}\textcolor{red}{d} \circ \text{zero} &= \lambda z. \lambda n. \text{zero}, \\ \textcolor{red}{d} \circ \text{succ} &= (\lambda f. \lambda n. \text{add}(f(n), n)) \circ \textcolor{red}{d}.\end{aligned}$$

Moreover,  $\text{sol}(\text{mul}) = \text{uncurry}(\text{sol}(d))$ .

6. ([69], Example 20) Let  $\text{fact} : \text{nat} \rightarrow \text{nat}$  be a variable for the factorial function. The equations

$$\begin{aligned}\text{fact}(\text{zero}(z)) &= \text{one}, \\ \text{fact}(\text{succ}(n)) &= \text{mul}(\text{fact}(n), \text{succ}(n))\end{aligned}$$

for  $\text{fact}$  entail the equations

$$\begin{aligned}d(\text{zero}(z)) &= (\text{one}, \text{zero}), \\ d(\text{succ}(k)) &= (\lambda(m, n).(\text{mul}(m, \text{succ}(n)), \text{succ}(n)))(d(k))\end{aligned}$$

for  $d =^{\mathcal{A}} \langle \text{fact}, \text{id}_{\text{nat}} \rangle : \text{nat} \rightarrow \text{nat} \times \text{nat}$ . An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned}\textcolor{red}{d} \circ \text{zero} &= \langle \text{one}, \text{zero} \rangle, \\ \textcolor{red}{d} \circ \text{succ} &= (\lambda(m, n).(\text{mul}(m, \text{succ}(n)), \text{succ}(n))) \circ \textcolor{red}{d}.\end{aligned}$$

Moreover,  $\text{sol}(\text{fact}) = \pi_1 \circ \text{sol}(d)$ . Hence  $\text{fact}$  is a paramorphism (see chapter 12).

A single equation (between  $\lambda\Sigma$ -terms) that defines  $\text{fact}$  (see chapter 14):

$$\textcolor{red}{fact} = \lambda n. \text{if } (= \text{zero})(n) \text{ then } \text{succ}(n) \text{ else } \text{mult}(n, \textcolor{red}{fact}(\text{pred}(n))). \quad (4)$$

7. Let  $sub : nat \times nat \rightarrow nat$  be a variable for subtraction on  $A_{Nat}$ . The equations

$$\begin{aligned} sub(zero(z), n) &= zero, \\ sub(succ(m), zero(z)) &= succ(m), \\ sub(succ(m), succ(n)) &= sub(m, n) \end{aligned}$$

for  $sub$  entail the equations

$$\begin{aligned} d(zero(z)) &= (\lambda n. zero, zero), \\ d(succ(k)) &= (\lambda(f, m). (case\{zero. \lambda z. succ(m), succ.f\}, succ(m))) (d(k)) \end{aligned}$$

for  $d =^{\mathcal{A}} \langle \text{curry}(sub), id_{nat} \rangle : nat \rightarrow (nat \rightarrow nat) \times nat$ . An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} \textcolor{red}{d} \circ zero &= \langle \lambda z. \lambda n. zero, zero \rangle, \\ \textcolor{red}{d} \circ succ &= (\lambda(f, m). (case\{zero. \lambda z. succ(m), succ.f\}, succ(m))) \circ \textcolor{red}{d}. \end{aligned}$$

Moreover,  $sol(sub) = uncurry(\pi_1 \circ sol(d))$ .

8. Let  $ack : nat \times nat \rightarrow nat$  be a variable for the Ackermann function. The equations

$$\begin{aligned} ack(zero(z), n) &= succ(n) \\ ack(succ(m), zero(z)) &= ack(m, one) \\ ack(succ(m), succ(n)) &= ack(m, ack(succ(m), n)) \end{aligned}$$

for  $ack$  entail the equations

$$d(zero(z)) = succ \tag{5}$$

$$d(succ(m))(zero(z)) = d(m)(one) \tag{6}$$

$$d(succ(m))(succ(n)) = d(m)(d(succ(m))(n)) \tag{7}$$

for  $d =^A curry(ack) : nat \rightarrow (nat \rightarrow nat)$ . According to [75], section 5.2, the nesting of  $d$ -applications in (7) suggest to first derive an inductive definition of the “inner call”  $d(succ(m))$ . From (6) and (7) we obtain

$$d(succ(m)) \circ zero = \lambda z. d(m)(one), \tag{8}$$

$$d(succ(m)) \circ succ = d(m) \circ d(succ(m)). \tag{9}$$

For all  $f : A_{nat} \rightarrow A_{nat}$ , let  $h_f : nat \rightarrow nat \in V$ . By Theorem RECFUN1, the equations

$$h_f \circ zero = \lambda z. f(one),$$

$$h_f \circ succ = f \circ h_f$$

have a unique solution  $sol(h_f)$  in  $h_f$ . Hence by (8) and (9),  $\mathcal{A}$  satisfies

$$h(d(m)) = d(succ(m)). \quad (10)$$

Let

$$\begin{aligned} h : (A_{nat} \rightarrow A_{nat}) &\rightarrow (A_{nat} \rightarrow A_{nat}) \\ f &\mapsto sol(h_f). \end{aligned}$$

(5) and (10) yield an inductive definition of  $d$ :

$$\begin{aligned} \textcolor{red}{d} \circ zero &= \lambda z. succ \\ \textcolor{red}{d} \circ succ &= \textcolor{blue}{h} \circ \textcolor{red}{d}. \end{aligned}$$

Moreover,  $sol(ack) = uncurry(sol(d))$ .

9. Let  $x : X$ ,  $f : X \rightarrow X^*$ ,  $g : X^* \rightarrow X^*$   $h : X^* \rightarrow X^* \in V$  and

$$\begin{aligned} replicate &: nat \rightarrow (X \rightarrow X^*), \\ take &: nat \rightarrow (X^* \rightarrow 1 + X^*), \\ takeStream &: nat \rightarrow (X^{\mathbb{N}} \rightarrow X^*) \end{aligned}$$

be variables for the synonymous Haskell functions. Inductive definitions of the three functions read as follows:

$$\begin{aligned} replicate \circ zero &= \lambda z. \lambda x. \epsilon, \\ replicate \circ succ &= (\lambda f. \lambda x. (x \cdot f(x))) \circ \textcolor{red}{replicate}, \end{aligned}$$

$$\text{take} \circ \text{zero} = \lambda z. \lambda x. \iota_2(\epsilon),$$

$$\text{take} \circ \text{succ} = (\lambda g. \lambda s. (\text{case}\{\alpha. \iota_1, \text{cons}. \lambda(x, s'). (x \cdot g(s'))\})^{X^*}) \circ \text{take},$$

$$\text{takeStream} \circ \text{zero} = \lambda z. \lambda x. \epsilon,$$

$$\text{takeStream} \circ \text{succ} = (\lambda h. \lambda s. (\text{head}^{\text{InfSeq}}(s) \cdot h(\text{tail}^{\text{InfSeq}}(s)))) \circ \text{takeStream}$$

(see sample algebras 3 and 5). The interpretation of the case distinction in  $X^*$  is well-defined because  $X^*$  is initial in  $\text{Alg}_{\text{List}(X)}$  (see above).

## Inductively defined functions on lists

Let  $X, Y, Z$  be sets and  $C\Sigma = \text{List}(X)$ . Suppose that  $\Sigma$  includes

$$\text{List}'(Y) = \text{List}(Y)[\text{state}'/\text{state}, \alpha'/\alpha, \text{cons}'/\text{cons}],$$

$$\text{List}''(Z) = \text{List}(Z)[\text{state}''/\text{state}, \alpha''/\alpha, \text{cons}''/\text{cons}]$$

(see chapter 8) and  $\mathcal{A}|_{\text{List}'(Y)}$  and  $\mathcal{A}|_{\text{List}''(Z)}$  are initial in  $\text{Alg}_{\text{List}'(Y)}$  and  $\text{Alg}_{\text{List}''(Z)}$ , respectively

The corresponding instance of (the single factor of) (1) and thus the schema for interpreting  $d : \text{state} \rightarrow e$  as an inductively defined function on  $A_{\text{List}(X)}$  reads as follows:

$$d \circ \alpha = \bar{\alpha}, \tag{11}$$

$$d \circ \text{cons} = \overline{\text{cons}} \circ (id_X \times d). \tag{12}$$

Let  $z : 1$ ,  $b : 2$ ,  $p : 2^{\mathbb{N}}$ ,  $x : X$ ,  $y : Y$ ,  $m, n : \mathbb{N}$ ,  $s, s' : state \in V$  and  $(\leq) : \mathbb{N}^2 \rightarrow 2$ ,  $(\wedge) : 2^2 \rightarrow 2$  be defined as usually.

**10.** Let  $d = length : state \rightarrow nat$  be a variable for the synonymous Haskell function with the equations

$$\begin{aligned} d(\alpha(z)) &= zero, \\ d(cons(x, s)) &= succ(d(s)). \end{aligned}$$

An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} d \circ \alpha &= zero, \\ d \circ cons &= succ \circ \pi_2 \circ (id_X \times d). \end{aligned}$$

**11.** Let  $X = \mathbb{N}$  and  $d = sum : state \rightarrow \mathbb{N}$  be a variable for the synonymous Haskell function with the equations

$$\begin{aligned} d(\alpha(z)) &= zero, \\ d(cons(n, s)) &= add(n, d(s)). \end{aligned}$$

An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} d \circ \alpha &= zero, \\ d \circ cons &= add \circ (id_{\mathbb{N}} \times d). \end{aligned}$$

12. Let  $conc : state \times state \rightarrow state$  be a variable for the concatenation of lists. The equations

$$\begin{aligned} conc(\alpha(z), s) &= s \\ conc(cons(x, s), s') &= cons(x, conc(s, s')) \end{aligned}$$

for  $conc$  entail the equations

$$\begin{aligned} d(\alpha(z))(s) &= s, \\ d(cons(x, s))(s') &= cons(x, d(s)(s')) \end{aligned}$$

for  $d =^{\mathcal{A}} curry(cons) : state \rightarrow (state \rightarrow state)$ . An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} \textcolor{red}{d} \circ \alpha &= \lambda z. id_{state}, \\ \textcolor{red}{d} \circ cons &= (\lambda(x, f). \lambda s'. cons(x, f(s'))) \circ (id_X \times \textcolor{red}{d}) \end{aligned}$$

where  $f : state \rightarrow state \in V$ . Moreover,  $sol(conc) = uncurry(sol(d))$ .

13. Let  $s' : state'$ ,  $f : Y^X$ ,  $g : Y^X$ ,  $h : Y^X \rightarrow state'$ ,  $h' : Z^{X \times Y} \rightarrow (state' \rightarrow state'') \in V$  and

$$\begin{aligned} map &: Y^X \rightarrow (state \rightarrow state'), \\ zipWith &: Z^{X \times Y} \rightarrow (state \rightarrow (state' \rightarrow state'')) \end{aligned}$$

be variables for the synonymous Haskell functions.

The equations

$$\begin{aligned}
 map(f)(\alpha(z)) &= \alpha', \\
 map(f)(cons(x, s)) &= cons'(f(x), map(f)(s)), \\
 zipWith(g)(\alpha(z))(s') &= \alpha'', \\
 zipWith(g)(s)(\alpha'(z)) &= \alpha'', \\
 zipWith(g)(cons(x, s))(cons(y, s')) &= cons''(g(x, y), zipWith(g)(s)(s'))
 \end{aligned}$$

for  $map$  and  $zipWith$  entail the equations

$$\begin{aligned}
 d(\alpha(z))(f) &= \alpha', \\
 d(cons(x, s))(f) &= cons'(f(x), d(s)(f)), \\
 d'(\alpha(z))(g)(s') &= \alpha'', \\
 d'(cons(x, s))(g)(\alpha'(z)) &= \alpha'', \\
 d'(cons(x, s))(g)(cons'(y, s')) &= cons''(g(x, y), d'(s)(g)(s')).
 \end{aligned}$$

for

$$\begin{aligned}
 d =^{\mathcal{A}} flip(map) : state \rightarrow (Y^X \rightarrow state'), \\
 d' =^{\mathcal{A}} flip(zipWith) : state \rightarrow (Z^{X \times Y} \rightarrow (state' \rightarrow state'')). 
 \end{aligned}$$

$\mathcal{A}$ -equivalent inductive definitions for  $d$  and  $d'$  read as follows:

$$\begin{aligned}
 \textcolor{red}{d} \circ \alpha &= \lambda z. \lambda f. \alpha', \\
 \textcolor{red}{d} \circ cons &= (\lambda(x, h). \lambda f. cons'(f(x), h(f))) \circ (id_X \times \textcolor{red}{d}),
 \end{aligned}$$

$$\begin{aligned}\textcolor{red}{d}' \circ \alpha &= \lambda z. \lambda g. \lambda s'. \alpha'', \\ \textcolor{red}{d}' \circ \text{cons} &= (\lambda(x, h'). \lambda g. \text{case}\{\alpha'. \alpha'', \text{cons}'. \lambda(y. s'). \text{cons}''(g(x, y), h'(g)(s'))\}) \\ &\quad \circ (id_X \times \textcolor{red}{d}').\end{aligned}$$

Moreover,  $\text{sol}(\text{map}) = \text{flip}(\text{sol}(d))$  and  $\text{sol}(\text{zipWith}) = \text{flip}(\text{sol}(d'))$ .

14. Let  $\text{filter} : 2^X \rightarrow (\text{state} \rightarrow \text{state})$  be a variable for the synonymous Haskell function and  $p : 2^X \in V$ . The equations

$$\begin{aligned}\text{filter}(p)(\alpha(z)) &= \alpha \\ \text{filter}(p)(\text{cons}(x, s)) &= \text{if } p(x) \text{ then } \text{cons}(x, \text{filter}(p)(s)) \text{ else } \text{filter}(p)(s)\end{aligned}$$

for  $\text{filter}$  entail the equations

$$\begin{aligned}d(\alpha(z))(p) &= \alpha, \\ d(\text{cons}(x, s))(p) &= \text{if } p(x) \text{ then } \text{cons}(x, d(s)(p)) \text{ else } d(s)(p)\end{aligned}$$

for  $d =^A \text{flip}(\text{filter}) : \text{state} \rightarrow (2^B \rightarrow \text{state})$ . An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned}\textcolor{red}{d} \circ \alpha &= \lambda z. \lambda p. \alpha, \\ \textcolor{red}{d} \circ \text{cons} &= (\lambda(x, f). \lambda p. \text{if } p(x) \text{ then } \text{cons}(x, f(p)) \text{ else } f(p)) \circ (id_X \times \textcolor{red}{d})\end{aligned}$$

where  $f : 2^X \rightarrow \text{state} \in V$ . Moreover,  $\text{sol}(\text{filter}) = \text{flip}(\text{sol}(d))$ .

15. Let  $foldl : Y^{Y \times X} \times Y \rightarrow (state \rightarrow Y)$  be a variable for the synonymous Haskell function and  $f : Y^{Y \times X} \in V$ . The equations

$$\begin{aligned} foldl(f, y)(\alpha(z)) &= y \\ foldl(f, y)(cons(x, s)) &= foldl(f, f(y, x))(s) \end{aligned}$$

for  $foldl$  entail the equations

$$\begin{aligned} d(\alpha(z))(f, y) &= y \\ d(cons(x, s))(f, y) &= d(s)(f, f(y, x)) \end{aligned}$$

for  $d =^A flip(foldl) : state \rightarrow (Y^{Y \times X} \times Y \rightarrow Y)$ .

An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} \textcolor{red}{d} \circ \alpha &= \lambda z. \pi_2, \\ \textcolor{red}{d} \circ cons &= (\lambda(x, g). \lambda(f, y). g(f, f(y, x))) \circ (id_X \times \textcolor{red}{d}) \end{aligned}$$

where  $g : Y^{Y \times X} \times Y \rightarrow Y \in V$ . Moreover,  $sol(foldl) = flip(sol(d))$ .

16. Let  $foldr : Y^{X \times Y} \times Y \rightarrow (state \rightarrow Y)$  be a variable for the synonymous Haskell function and  $f : Y^{X \times Y} \in V$ . The equations

$$\begin{aligned} foldr(f, y)(\alpha(z)) &= y \\ foldr(f, y)(cons(x, s)) &= f(x, foldr(f, y)(s)) \end{aligned}$$

for  $foldr$  entail the equations

$$\begin{aligned} d(\alpha(z))(f, y) &= y \\ d(cons(x, s))(f, y) &= f(x, d(s)(f, y)) \end{aligned}$$

for  $d =^{\mathcal{A}} flip(foldr) : state \rightarrow (Y^{X \times Y} \times Y \rightarrow Y)$ .

An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} d \circ \alpha &= \lambda z. \pi_2, \\ d \circ cons &= (\lambda(x, g). \lambda(f, y). f(x, g(f, y))) \circ (id_X \times d) \end{aligned}$$

where  $g : Y^{X \times Y} \times Y \rightarrow Y \in V$ . Moreover,  $sol(foldr) = flip(sol(d))$ .

By the way, for all  $List(X)$ -algebras  $\mathcal{B}$ ,

$$sol(foldr)(cons^{\mathcal{B}}, \alpha^{\mathcal{B}}) = fold^{\mathcal{B}}$$

(see chapter 9).

17. Let  $X = \mathbb{N}$  and  $\text{sorted} : \text{state} \rightarrow 2$  be a variable for the test on the sortedness of a list of natural numbers. The equations

$$\begin{aligned}\text{sorted}(\alpha(z)) &= 1 \\ \text{sorted}(\text{cons}(m, \alpha(z))) &= 1 \\ \text{sorted}(\text{cons}(m, \text{cons}(n, s))) &= m \leq n \wedge \text{sorted}(\text{cons}(n, s))\end{aligned}$$

for  $\text{sorted}$  entail the equations

$$\begin{aligned}d(\alpha(z)) &= \lambda n.1 \\ d(\text{cons}(m, s)) &= (\lambda p. \lambda n. m \leq n \wedge p(n))(d(s))\end{aligned}$$

for  $d =^{\mathcal{A}} \lambda s. \lambda n. \text{sorted}(\text{cons}(n, s)) : \text{state} \rightarrow 2^{\mathbb{N}}$ . An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned}\textcolor{red}{d} \circ \alpha &= \lambda z. \lambda n. 1, \\ \textcolor{red}{d} \circ \text{cons} &= (\lambda(m, p). \lambda n. m \leq n \wedge p(n)) \circ (id_{\mathbb{N}} \times \textcolor{red}{d}).\end{aligned}$$

From these equations and the first equation for  $\text{sorted}$  we obtain an inductive definition of  $\textcolor{red}{d}' =^{\mathcal{A}} \langle \text{sorted}, id_{\text{state}} \rangle$ :

$$\begin{aligned}\textcolor{red}{d}' \circ \alpha &= \langle \lambda z. 1, \alpha \rangle \\ \textcolor{red}{d}' \circ \text{cons} &= (\lambda(m, (b, s)). (d(s)(m), \text{cons}(m, s))) \circ (id_{\mathbb{N}} \times \textcolor{red}{d}').\end{aligned}$$

Moreover,  $\text{sol}(\text{sorted}) = \pi_1 \circ \text{sol}(\textcolor{blue}{d}')$ . Hence  $\text{sorted}$  is a paramorphism (see chapter 12).

18. Suppose that  $\Sigma$  includes  $Bintree(L)$  and  $\mathcal{A}_{Bintree(L)} = FBin(L)$  (see sample algebra 10). Let  $X = 2$  and  $d = subtree : state \rightarrow (btree \rightarrow btree)$  be a variable for the function that maps a node  $s$  of a finite binary tree  $t$  with node labels from  $L$  to the subtree of  $t$  with root  $s$

Let  $b : 2$ ,  $lab : L$ ,  $t, u, u' : btree$ ,  $f : btree \rightarrow btree \in V$ .

$d$  may be specified by the conditional equations

$$d(\alpha(z))(t) = t, \tag{1}$$

$$d(cons(b, s))(bjoin(x, t, u)) = \text{if } b = 0 \text{ then } d(s)(t) \text{ else } d(s)(u), \tag{2}$$

$$d(cons(b, s))(empty) = empty. \tag{3}$$

The conjunction of (2) and (3) is  $\mathcal{A}$ -equivalent to the equation

$$\begin{aligned} d(cons(b, s)) = & \text{case}\{bjoin.\lambda(lab, t, u).\text{if } b = 0 \text{ then } d(s)(t) \text{ else } d(s)(u), \\ & empty.\lambda z.empty\}. \end{aligned}$$

Hence an  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} d \circ \alpha &= \lambda z.id_{btree}, \\ d \circ cons &= (\lambda(b, f).\text{case}\{bjoin.\lambda(lab, t, u).\text{if } b = 0 \text{ then } f(t) \text{ else } f(u), \\ &\quad empty.\lambda z.empty\}) \circ (id_2 \times d). \end{aligned}$$

19. Suppose that  $\Sigma$  includes  $coBintree(L)$  and  $\mathcal{A}_{coBintree(L)} = Bin(L)$  (see sample algebra 12). Let  $X = 2$  and  $d = subtreeC : state \rightarrow (btree \rightarrow btree)$  be a variable for the function that maps a node  $s$  of a finite or infinite binary tree  $t$  with node labels from  $L$  to the subtree of  $t$  with root  $s$

Let  $b : 2$ ,  $lab : L$ ,  $t, u, u' : btree$ ,  $f : btree \rightarrow btree \in V$ .

$d$  may be specified by the conditional equations

$$d(\alpha(z))(t) = t, \quad (4)$$

$$d(cons(b, s))(t) = \text{if } b \text{ then } d(s)(u') \text{ else } d(s)(u) \Leftarrow split(t) = \iota_1(lab, u, u'), \quad (5)$$

$$d(cons(b, s))(t) = t \Leftarrow split(t) = \iota_2. \quad (6)$$

The conjunction of (5) and (6) is  $\mathcal{A}$ -equivalent to the equation

$$d(cons(b, s)) = \lambda t. [\lambda(lab, u, u'). \text{if } b \text{ then } d(s)(u') \text{ else } d(s)(u), \lambda z. t](split(t)).$$

Hence an  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned} \textcolor{red}{d} \circ \alpha &= \lambda z. id_{btree}, \\ \textcolor{red}{d} \circ cons &= (\lambda(b, f). \lambda t. [\lambda(lab, u, u'). \text{if } b \text{ then } f(u') \text{ else } f(u), \lambda z. t](split(t))) \\ &\quad \circ (id_2 \times \textcolor{red}{d}). \end{aligned}$$

## Further inductively defined functions

### 20. The Brzozowski automaton

Let  $C\Sigma = \text{Reg}(X)$  (see chapter 8).

Let  $z : 1, b, c : 2, x : X, t, u : \text{state}, f, g : \text{state}^X \in V, B \subseteq X$  and  $(\in) : X \times \mathcal{P}(X) \rightarrow 2$  and  $\text{max}, (*) : 2 \times 2 \rightarrow 2$  be defined as usually.

The equations for  $\delta : \text{state} \rightarrow \text{state}^X$  given in sample algebra 23 entail the equations

$$\begin{aligned}
 d(\text{par}(t, u)) &= (\lambda((f, t), (g, u)).(\lambda x.\text{par}(f(x), g(x)), \text{par}(t, u))) \\
 &\quad (d(t), d(u)), \\
 d(\text{seq}(t, u)) &= (\lambda((f, t), (g, u)).(\lambda x.\text{if } \beta(t) \text{ then } \text{par}(\text{seq}(f(x), u), g(x)) \\
 &\quad \text{else } \text{seq}(f(x), u), \text{seq}(t, u))) \\
 &\quad (d(t), d(u)), \\
 d(\text{iter}(t)) &= (\lambda(f, t).(\lambda x.f(x), \text{iter}(t)))(d(t)), \\
 d(\text{eps}(z)) &= (\lambda x.\text{base}(\emptyset), \text{eps}), \\
 d(\text{base}(B)) &= (\lambda x.\text{if } x \in B \text{ then } \text{eps} \text{ else } \text{base}(\emptyset), \text{base}(B))
 \end{aligned}$$

for  $d = ^{\mathcal{A}} \langle \delta, \text{id}_{\text{state}} \rangle : \text{state} \rightarrow \text{state}^X \times \text{state}$ .

An  $\mathcal{A}$ -equivalent inductive definition of  $d$  reads as follows:

$$\begin{aligned}
 d \circ par &= (\lambda((f, t), (g, u)).(\lambda x. par(f(x), g(x)), par(t, u))) \circ (d \times d), \\
 d \circ seq &= (\lambda((f, t), (g, u)).(\lambda x. if \beta(t) then par(seq(f(x), u), g(x)) \\
 &\quad else seq(f(x), u)), seq(t, u)) \circ (d \times d), \\
 d \circ iter &= (\lambda(f, t).(\lambda x. f(x), iter(t))) \circ d, \\
 d \circ eps &= \lambda z.(\lambda x. base(\emptyset), eps), \\
 d \circ base &= \lambda B.(\lambda x. if x \in B then eps else base(\emptyset), base(B)).
 \end{aligned}$$

Moreover,  $sol(\delta) = \pi_1 \circ sol(d)$ . Hence  $\delta$  is a paramorphism (see chapter 12). An inductive definition of  $\beta : state \rightarrow 2$  that is  $\mathcal{A}$ -equivalent to the equations for  $\beta$  given in sample algebra 23 reads as follows:

$$\begin{aligned}
 \beta \circ par &= (\lambda(b, c). max(b, c)) \circ (\beta \times \beta), \\
 \beta \circ seq &= (\lambda(b, c). b * c) \circ (\beta \times \beta), \\
 \beta \circ iter &= (\lambda b. 1) \circ \beta, \\
 \beta \circ eps &= \lambda z. 1, \\
 \beta \circ base &= \lambda B. 0.
 \end{aligned}$$

## 21. Balanced binary trees ([53], Example 4.12)

Let  $X$  be a set and  $C\Sigma = \text{Bintree}(X)$  (see chapter 8).

Let  $z : 1, x : X, m, n : \mathbb{N}, b, c : 2, t, u : \text{btree} \in V$  and

$$\max : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}, \quad (+1) : \mathbb{N} \rightarrow \mathbb{N}, \quad (=) : \mathbb{N} \times \mathbb{N} \rightarrow 2, \quad (*) : 2 \times 2 \rightarrow 2$$

be defined as usually and

$$d = \langle \text{height}, \text{bal} \rangle : \text{btree} \rightarrow \mathbb{N} \times 2$$

be a variable for the function that maps a finite binary tree  $t$  with node labels from  $X$  to both the height of  $t$  and a Boolean value that indicates whether  $t$  is balanced or not (see sample algebra 10).

Equations for  $d$  read as follows:

$$\begin{aligned} d(\text{empty}(z)) &= (0, \text{True}), \\ d(\text{bjoin}(x, t, u)) &= (\lambda((m, b), (n, c)).(\max(m, n) + 1, b * c * (m = n)))(d(t), d(u)). \end{aligned}$$

They are  $\mathcal{A}$ -equivalent to the following inductive definition of  $d$ :

$$\begin{aligned} d \circ \text{bjoin} &= (\lambda(x, (m, b), (n, c)).(\max(m, n) + 1, b * c * (m = n))) \circ (id_X \times d \times d), \\ d \circ \text{empty} &= \lambda z.(0, \text{True}). \end{aligned}$$

Moreover,  $\text{sol}(\text{height}) = \pi_1 \circ \text{sol}(d)$  and  $\text{sol}(\text{bal}) = \pi_2 \circ \text{sol}(d)$ .

## 22. Flatten a finite tree ([69], Example 4)

Let  $X$  be a set and  $C\Sigma = \text{Tree}(X)$ . Suppose that  $\Sigma$  also includes  $\text{List}(X)$  (see chapter 8).

Let  $z : 1$ ,  $x : X$ ,  $t : \text{tree}$ ,  $ts : \text{trees} \in V$  and  $\text{conc} : \text{state} \times \text{state} \rightarrow \text{state}$  be interpreted as in example 12 above and

$$d_{\text{tree}} = \text{flatten} : \text{tree} \rightarrow \text{list}, \quad d_{\text{trees}} = \text{flattenL} : \text{trees} \rightarrow \text{list}$$

be variables for the functions that map a (list of) finite tree(s) to its list of the node labels in depthfirst order (see sample algebra 13).

Equations for  $\text{flatten}$  and  $\text{flattenL}$  read as follows:

$$\begin{aligned} \text{flatten}(\text{join}(x, ts)) &= \text{cons}(x, \text{flattenL}(ts)), \\ \text{flattenL}(\alpha(z)) &= \alpha, \\ \text{flattenL}(\text{cons}(t, ts)) &= \text{conc}(\text{flatten}(t), \text{flattenL}(ts)). \end{aligned}$$

They are  $\mathcal{A}$ -equivalent to the following inductive definition of  $\{d_s \mid s \in \{\text{tree}, \text{trees}\}\}$ :

$$\begin{aligned} \text{flatten} \circ \text{join} &= \text{cons} \circ (\text{id}_X \times \text{flattenL}), \\ \text{flattenL} \circ \alpha &= \alpha, \\ \text{flattenL} \circ \text{cons} &= \text{conc} \circ (\text{flatten} \times \text{flattenL}). \end{aligned}$$

## Sample coinductively defined functions

Given a destructive polynomial subsignature  $D\Sigma = (S, \mathcal{I}, D)$  of  $\Sigma$  and a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  such that  $\mathcal{A}|_{D\Sigma}$  is final in  $Alg_{D\Sigma}$ , Theorem RECFUN2 tells us that a set of coinductively defined functions can be regarded as the unique solution

$$\{\text{sol}(c_s) : A_{e_s} \rightarrow A_s \mid s \in S\}$$

of the formula

$$\bigwedge_{d:s \rightarrow e \in D} d \circ c_s = C_e \circ \bar{d} \quad (1)$$

in the set  $C = \{c_s : e_s \rightarrow s \mid s \in S\}$  of  $\lambda\Sigma$ -term variables.

In each of the following examples, we start out from a given set  $E$  of equations whose conjunction is equivalent to (1), i.e., (1) is solvable in  $C$  iff  $E$  is solvable in the variables of  $E$ .

Often the solution of (1) is a sum extension of the solution of  $E$  and a further function  $f$ . For instance, if  $f$  is an identity, then  $E$  specifies a primitive corecursive function or apomorphism (see chapter 12).

If the equations for  $C$  are given in some applicative form, “recursive calls” of  $C$  may scatter around on the equations’ right-hand sides. In order to obtain the right-hand side of (1), they must be bundled into a single term  $C_e : e[e_s/s \mid s \in S] \rightarrow e$ .

## Coinductively defined functions to natural numbers with infinity

Let  $D\Sigma = coNat$  (see chapter 8). The corresponding instance of (the single factor of) (1) and thus the schema for interpreting  $c : e \rightarrow nat$  as a coinductively defined function to  $A_{coNat}$  reads as follows:

$$pred \circ c = (id_1 + c) \circ \overline{pred}. \quad (2)$$

Let  $z : 1, m, m', n, n' : nat \in V$ .

1. Let  $c = zero : 1 \rightarrow nat$  be a variable for number 0 with the equation

$$pred(c(z)) = \iota_1.$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$pred \circ c = \iota_1 = \iota_1 \circ id_1 \stackrel{(*)}{=} (id_1 + c) \circ \iota_1.$$

(\*) is an instance of equation (18) in section [Sums](#).

2. Let  $c = infinity : 1 \rightarrow nat$  be a variable for the ordinal number  $\omega$  with the equation

$$pred(c(z)) = \iota_2(c).$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$pred \circ \textcolor{red}{c} = \iota_2 \circ c \stackrel{(*)}{=} (id_1 + \textcolor{red}{c}) \circ \iota_2.$$

(\*) is an instance of equation (18) in section [Sums](#).

3. The successor function on  $A_{coNat}$  is defined non-recursively:

$$succ = obj\{pred.\iota_2\} : nat \rightarrow nat.$$

4. Suppose that  $\Sigma$  includes  $coList(X)$  (see chapter 8) and  $\mathcal{A}|_{Colist(X)}$  is final in  $Alg_{coList(X)}$ . Let  $x : X$ ,  $s, s' : state \in V$  and  $\textcolor{red}{c} = \text{length} : state \rightarrow nat$  be a variable for the function that computes the length of a colist. The conjunction of the conditional equations

$$\begin{aligned} pred(c(s)) &= \iota_2(c(s')) \Leftarrow split(s) = \iota_1(x, s'), \\ pred(c(s)) &= \iota_1 \Leftarrow split(s) = \iota_2 \end{aligned}$$

is  $\mathcal{A}$ -equivalent to the unconditional equation

$$pred(c(s)) = [\lambda(x, s').\iota_2(c(s')), \lambda z.\iota_1](split(s)).$$

Hence an  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$pred \circ \textcolor{red}{c} = (id_1 + \textcolor{red}{c}) \circ [\lambda(x, s').\iota_2(s'), \lambda z.\iota_1] \circ split.$$

5. ([81], Example 2.6.6) Let  $c = add : nat \times nat \rightarrow nat$  be a variable for addition on  $A_{coNat}$ . The conditional equations

$$pred(c(m, n)) = \iota_1 \Leftarrow pred(m) = \iota_1 \wedge pred(n) = \iota_1,$$

$$pred(c(m, n)) = \iota_2(c(m, n')) \Leftarrow pred(m) = \iota_1 \wedge pred(n) = \iota_2(n'),$$

$$pred(c(m, n)) = \iota_2(c(m', n)) \Leftarrow pred(m) = \iota_2(m')$$

entail the single unconditional equation

$$pred(c(m, n)) = [\lambda z. [\lambda z. \iota_1, \lambda n'. \iota_2(c(m, n'))](pred(n)), \lambda m'. \iota_2(c(m', n))](pred(m)).$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} pred \circ c &= (id_1 + c) \circ \\ &\quad \lambda(m, n). [\lambda z. [\iota_1, \lambda n'. \iota_2(m, n')](pred(n)), \lambda m'. \iota_2(m', n)](pred(m)). \end{aligned}$$

## Coinductively defined functions to streams

Let  $X, Y$  be sets and  $D\Sigma = \text{Stream}(X)$ . Suppose that

$$\text{Stream}'(Y) = \text{Stream}(Y)[\text{state}'/\text{state}, \text{head}'/\text{head}, \text{tail}'/\text{tail}],$$

$$\text{Stream}''(Z) = \text{Stream}(Z)[\text{state}''/\text{state}, \text{head}''/\text{head}, \text{tail}''/\text{tail}]$$

(see chapter 8) and  $\mathcal{A}|_{\text{Stream}'(Y)}$  and  $\mathcal{A}|_{\text{Stream}''(Z)}$  are final in  $\text{Alg}_{\text{Stream}'(Y)}$  and  $\text{Alg}_{\text{Stream}''(Z)}$ , respectively. The corresponding instance of (the single factor of) (1) and thus the schema for interpreting  $c : e \rightarrow \text{state}$  as a coinductively defined function into  $A_{\text{Stream}(X)}$  reads as follows:

$$\text{head} \circ c = \overline{\text{head}}, \quad (3)$$

$$\text{tail} \circ c = c \circ \overline{\text{tail}}. \quad (4)$$

Let  $z : 1, x : X, y : Y, m, n : \mathbb{N}, s, s' : \text{state} \in V$  and  $(+), \min : \mathbb{R}^2 \rightarrow \mathbb{R}, (=), (<) : \mathbb{R}^2 \rightarrow 2, (\wedge) : 2^2 \rightarrow 2$  be defined as usually.

6. ([167], Example 4) Let  $X = \mathbb{N}$  and  $c = \text{nats} : \mathbb{N} \rightarrow \text{state}$  be a variable for the function that maps  $n \in \mathbb{N}$  to the ordered stream of all natural numbers  $\geq n$  with the equations

$$\text{head}(c(n)) = n$$

$$\text{tail}(c(n)) = c(n + 1).$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} \text{head} \circ c &= id_{\mathbb{N}}, \\ \text{tail} \circ c &= c \circ (+1). \end{aligned}$$

7. Let  $c = \text{evens} : \text{state} \rightarrow \text{state}$  be a variable for the function that, given a stream  $s$ , returns the stream of all elements of  $s$  at even positions. Equations for  $\text{evens}$  read as follows:

$$\begin{aligned} \text{head}(c(s)) &= \text{head}(s) \\ \text{tail}(c(s)) &= c(\text{tail}(\text{tail}(s))). \end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} \text{head} \circ c &= \text{head}, \\ \text{tail} \circ c &= c \circ \text{tail} \circ \text{tail}. \end{aligned}$$

8. Let  $c = \text{zip} : \text{state} \times \text{state} \rightarrow \text{state}$  be a variable for the function to  $A_{\text{Stream}(X)}$  with the equations

$$\begin{aligned} \text{head}(c(s, s')) &= \text{head}(s) \\ \text{tail}(c(s, s')) &= c(s', \text{tail}(s)). \end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} \text{head} \circ c &= \text{head} \circ \pi_1, \\ \text{tail} \circ c &= c \circ \langle \pi_2, \text{tail} \circ \pi_1 \rangle. \end{aligned}$$

9. Let  $X = 2$  and  $blink, blink' : 1 \rightarrow state$  be variables for the Boolean streams with the equations

$$\begin{aligned} head(blink) &= 0 \\ tail(blink) &= blink' \\ head(blink') &= 1 \\ tail(blink') &= blink. \end{aligned}$$

They entail the equations

$$\begin{aligned} head(c(\iota_1(z))) &= 0, \\ head(c(\iota_2(z))) &= 1, \\ tail(c(\iota_1(z))) &= c(\iota_2(z)), \\ tail(c(\iota_2(z))) &= c(\iota_1(z)) \end{aligned}$$

for  $c =^{\mathcal{A}} [blink, blink'] : 1 + 1 \rightarrow state$ . An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} head \circ c &= [\lambda z.0, \lambda z.1], \\ tail \circ c &= c \circ [\iota_2, \iota_1]. \end{aligned}$$

Moreover,  $sol(blink) = sol(c) \circ \iota_1$  and  $sol(blink') = sol(c) \circ \iota_2$ .

10. Let  $X = \mathbb{R}$  and  $c = appzeros : \mathbb{R} \rightarrow state$  be a variable for the function that appends a real number to the stream of zeros, specified by the equations

$$\begin{aligned} head(c(x)) &= x, \\ tail(c(x)) &= c(0). \end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} head \circ c &= id_{\mathbb{R}}, \\ tail \circ c &= c \circ \lambda x.0. \end{aligned}$$

11. ([26], Example 2.5; [146], Theorem 3.1) Let  $X = \mathbb{R}$  and

$$c = add : state \times state \rightarrow state$$

be a variable for addition on  $A_{Stream(X)}$  with the equations

$$\begin{aligned} head(c(s, s')) &= head(s) + head(s'). \\ tail(c(s, s')) &= c(tail(s), tail(s')). \end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} head \circ c &= \lambda(s, s').head(s) + head(s'), \\ tail \circ c &= c \circ (tail \times tail). \end{aligned}$$

12. ([167], Example 14 - a *futumorphism*) Let  $\text{exchange} : \text{state} \rightarrow \text{state}$  be a variable for the function on  $A_{\text{Stream}(X)}$  with the equations

$$\begin{aligned}\text{head}(\text{exchange}(s)) &= \text{head}(\text{tail}(s)), \\ \text{head}(\text{tail}(\text{exchange}(s))) &= \text{head}(s), \\ \text{tail}(\text{tail}(\text{exchange}(s))) &= \text{exchange}(\text{tail}(\text{tail}(s))).\end{aligned}$$

They entail the equations

$$\begin{aligned}\text{head}(c(\iota_1(s))) &= \text{head}(\text{tail}(s)), \\ \text{head}(c(\iota_2(s))) &= \text{head}(s), \\ \text{tail}(c(\iota_1(s))) &= c(\iota_2(s)), \\ \text{tail}(c(\iota_2(s))) &= c(\iota_1(\text{tail}(\text{tail}(s))))\end{aligned}$$

for  $c =^{\mathcal{A}} [\text{exchange}, \text{tail} \circ \text{exchange}] : \text{state} + \text{state} \rightarrow \text{state}$ . An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned}\text{head} \circ c &= [\text{head} \circ \text{tail}, \text{head}], \\ \text{tail} \circ c &= c \circ [\iota_2, \iota_1 \circ \text{tail} \circ \text{tail}].\end{aligned}$$

Moreover,  $\text{sol}(\text{exchange}) = \text{sol}(c) \circ \iota_1$ .

13. ([69], Example 10) Let  $X = \mathbb{N}$  and  $nats\&squares : \mathbb{N} \rightarrow state$  be a variable for the function that maps  $n \in \mathbb{N}$  to the ordered stream of all natural numbers  $\geq n$ , interleaved with the ordered stream of squares  $\geq n$ . Equations for  $nats\&squares$  read as follows:

$$\begin{aligned} head(nats\&squares(n)) &= n, \\ head(tail(nats\&squares(n))) &= n * n, \\ tail(tail(nats\&squares(n))) &= nats\&squares(n + 1). \end{aligned}$$

They entail the equations

$$\begin{aligned} head(c(\iota_1(n))) &= n, \\ head(c(\iota_2(n))) &= n * n, \\ tail(c(\iota_1(n))) &= c(\iota_2(n)), \\ tail(c(\iota_2(n))) &= c(\iota_1(n + 1)) \end{aligned}$$

for  $c =^{\mathcal{A}} [nats\&squares, tail \circ nats\&squares] : \mathbb{N} + \mathbb{N} \rightarrow state$ . An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} head \circ c &= [id_{\mathbb{N}}, \lambda n. n * n], \\ tail \circ c &= c \circ [\iota_2, \iota_1 \circ (+1)]. \end{aligned}$$

Moreover,  $sol(nats\&squares) = sol(c) \circ \iota_1$ .

14. Let  $c = \text{iterate} : X^X \times X \rightarrow \text{state}$  be a variable for the synonymous Haskell function with the equations

$$\begin{aligned}\text{head}(c(f, x)) &= x, \\ \text{tail}(c(f, x)) &= c(f, f(x)).\end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned}\text{head} \circ c &= \pi_2, \\ \text{tail} \circ c &= c \circ \lambda(f, x). (f, f(x)).\end{aligned}$$

15. ([37], Example 2.3) Let  $s' : \text{state}'$ ,  $f : Y^X$ ,  $g : Z^{X \times Y} \in V$  and

$$\begin{aligned}c &= \text{map} : Y^X \times \text{state} \rightarrow \text{state}', \\ c' &= \text{zipWith} : Z^{X \times Y} \times \text{state} \times \text{state}' \rightarrow \text{state}''\end{aligned}$$

be variables for the synonymous Haskell functions with the equations

$$\begin{aligned}\text{head}'(c(f, s)) &= f(\text{head}(s)), \\ \text{tail}'(c(f, s)) &= c(f, \text{tail}(s)), \\ \text{head}''(c'(g, s, s')) &= g(\text{head}(s), \text{head}'(s')), \\ \text{tail}''(c'(g, s, s')) &= c'(g, \text{tail}(s), \text{tail}'(s')).\end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} \text{head}' \circ \textcolor{red}{c} &= \lambda(f, s).f(\text{head}(s)), \\ \text{tail}' \circ \textcolor{red}{c} &= \textcolor{red}{c} \circ \lambda(f, s).(f, \text{tail}(s)), \\ \text{head}'' \circ \textcolor{red}{c}' &= \lambda(g, s, s').g(\text{head}(s), \text{head}(s')), \\ \text{tail}'' \circ \textcolor{red}{c}' &= \textcolor{red}{c} \circ \lambda(g, s, s').(g, \text{tail}(s), \text{tail}(s')). \end{aligned}$$

16. Let  $f : Y^{Y \times X} \in V$  and  $\textcolor{red}{c} = \text{foldprefixes} : Y^{Y \times X} \times Y \times \text{state} \rightarrow \text{state}'$  be a variable for the function with the equations

$$\begin{aligned} \text{head}(c(f, y, s)) &= f(y, \text{head}(s)), \\ \text{tail}(c(f, y, s)) &= c(f, f(y, \text{head}(s)), \text{tail}(s)). \end{aligned}$$

Hence, if  $A_{\text{state}} = X^{\mathbb{N}}$  and  $A_{\text{state}'} = Y^{\mathbb{N}}$ , then  $\textcolor{blue}{c}(f, y, s) = \lambda n.\text{foldl}(f, y)(\text{take}(n)(s))$ . An instance of *foldprefixes* is given by [37], Example 2.5. An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} \text{head} \circ \textcolor{red}{c} &= \lambda(f, y, s).f(y, \text{head}(s)), \\ \text{tail} \circ \textcolor{red}{c} &= \textcolor{red}{c} \circ \lambda(f, y, s).(f, f(y, \text{head}(s)), \text{tail}(s)). \end{aligned}$$

17. Suppose that  $\Sigma$  includes  $Nat$  and  $\mathcal{A}|_{Nat}$  is initial in  $Alg_{Nat}$ . Let  $cycleNats : nat \rightarrow state$  be a variable for the function that maps  $n \in \mathbb{N}$  to the stream of repetitions of the list  $[n, n - 1, \dots, 0]$  (see [1], section 2.1). Together with an auxiliary function  $c : nat \times nat \rightarrow state$ , it may be specified by the equations

$$\begin{aligned} cycleNats(n) &= c(n, n), \\ head(c(m, n)) &= m, \\ tail(c(zero, n)) &= c(n, n), \\ tail(c(succ(m), n)) &= c(m, n). \end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} head \circ c &= \pi_1, \\ tail \circ c &= c \circ \lambda(m, n).case\{zero.\lambda z.(n, n), succ.\lambda m.(m, n)\}(m). \end{aligned}$$

18. ([37], Example 2.2) Let  $X = \mathbb{R}$  and  $c = merge : state \times state \rightarrow state$  be a variable for the function with the equations

$$\begin{aligned} head(c(s, s')) &= min(head(s), head(s')), \\ tail(c(s, s')) &= if head(s) < head(s') then c(tail(s), s') \\ &\quad else if head(s) = head(s') then c(tail(s), tail(s')) else c(s, tail(s')). \end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} \text{head} \circ c &= \lambda(s, s'). \min(\text{head}(s), \text{head}(s')), \\ \text{tail} \circ c &= c \circ \lambda(s, s'). \text{if } \text{head}(s) < \text{head}(s') \text{ then } (\text{tail}(s), s') \\ &\quad \text{else if } \text{head}(s) = \text{head}(s') \text{ then } (\text{tail}(s), \text{tail}(s')) \\ &\quad \text{else } (s, \text{tail}(s')). \end{aligned}$$

19. ([167], Example 10) Let  $X = \mathbb{R}$  and  $\text{insert} : \mathbb{R} \times \text{state} \rightarrow \text{state}$  be a variable for the function with the equations

$$\begin{aligned} \text{head}(\text{insert}(x, s)) &= \min(x, \text{head}(s)), \\ \text{tail}(\text{insert}(x, s)) &= \text{if } x \leq \text{head}(s) \text{ then } s \text{ else } \text{insert}(x, \text{tail}(s)) \end{aligned}$$

They entail the equations

$$\begin{aligned} \text{head}(c(\iota_1(x, s))) &= \min(x, \text{head}(s)), \\ \text{head}(c(\iota_2(s))) &= \text{head}(s), \\ \text{tail}(c(\iota_1(x, s))) &= \text{if } x \leq \text{head}(s) \text{ then } c(\iota_2(s)) \text{ else } c(\iota_1(x, \text{tail}(s))), \\ \text{tail}(c(\iota_2(s))) &= c(\iota_2(\text{tail}(s))) \end{aligned}$$

for  $c =^{\mathcal{A}} [\text{insert}, \text{id}] : (\mathbb{R} \times \text{state}) + \text{state} \rightarrow \text{state}$ . An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} \text{head} \circ c &= [\lambda(x, s). \min(x, \text{head}(s)), \text{head}], \\ \text{tail} \circ c &= c \circ [\lambda(x, s). \text{if } x \leq \text{head}(s) \text{ then } \iota_2(s) \text{ else } \iota_1(x, \text{tail}(s)), \iota_2 \circ \text{tail}]. \end{aligned}$$

Moreover,  $sol(insert) = sol(c) \circ \iota_1$ .

## Coinductively defined functions to colists

Let  $X, Y$  be sets and  $D\Sigma = coList(X)$ . Suppose that

$$\begin{aligned} coList'(Y) &= coList(Y)[state'/state, split'/split], \\ coList''(Z) &= coList(Z)[state''/state, split''/split] \end{aligned}$$

(see chapter 8) and  $\mathcal{A}|_{coList'(Y)}$  and  $\mathcal{A}|_{coList''(Z)}$  are final in  $Alg_{coList'(Y)}$  and  $Alg_{coList''(Z)}$ , respectively. The corresponding instance of (the single factor of) (1) and thus the schema for interpreting  $c : e \rightarrow state$  as a coinductively defined function into  $A_{coList(X)}$  reads as follows:

$$split \circ c = (id_X \times c + id_1) \circ \overline{split}. \quad (5)$$

Let  $z : 1$ ,  $x : X$ ,  $y : Y$ ,  $m, n : \mathbb{N}$ ,  $s, s' : state \in V$  and  $(+) : \mathbb{R}^2 \rightarrow \mathbb{R}$ ,  $(=), (<) : \mathbb{R}^2 \rightarrow 2$ ,  $(\wedge) : 2^2 \rightarrow 2$  be defined as usually.

**20.** ([81], Example 2.6.5) Let  $c = nil : 1 \rightarrow state$  be a variable for the empty list with the equation

$$split(nil(z)) = \iota_2.$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$split \circ c = (id_X \times c + id_1) \circ \iota_2.$$

21. ([81], Example 2.6.5) Suppose that  $\Sigma$  includes  $List'(X) = List(X)[state'/state]$  (see chapter 8) and  $\mathcal{A}|_{List'(X)}$  is initial in  $Alg_{List'(X)}$ . Let  $x : X$ ,  $s : state \in V$  and  $c = inc : state \times state \rightarrow state$  be a variable for the inclusion of  $A_{List'(X)}$  in  $A_{coList(X)}$  with the equations

$$\begin{aligned} split(inc(cons(x, s))) &= \iota_1(x, inc(s)), \\ split(inc(\alpha)) &= \iota_2. \end{aligned}$$

An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$split \circ c = (id_X \times c + id_1) \circ case\{cons.\iota_1, \alpha.\iota_2\}.$$

22. ([81], Example 2.6.5) \*\*\*\* Let  $s, s' : state \in V$  and  $c = conc : state \rightarrow state$  be a variable for the concatenation of two colists with the equations

$$conc : X^\infty \times X^\infty \rightarrow X^\infty$$

$$\begin{aligned} split(conc(s, s')) &= \lambda\{(\iota_1, \iota_1).\iota_1, \\ &\quad (\iota_1, \iota_2(x, s'')).\iota_2(x, conc(s, s'')), \\ &\quad (\iota_2(x, s''), z).\iota_2(x, conc(s'', s'))\}(split(s), split(s')) \end{aligned}$$

$$conc_{1+X \times list} : 1 + X \times (X^\infty \times X^\infty) \rightarrow 1 + X \times X^\infty$$

$$\begin{aligned} conc &= obj\{split.(id_1 + id_X \times conc) \\ &\quad \circ \lambda(s, s').\lambda\{(\iota_1, \iota_1).\iota_1, \\ &\quad (\iota_1, \iota_2(x, s'')).\iota_2(x, (s, s'')), \\ &\quad (\iota_2(x, s''), z).\iota_2(x, (s'', s'))\}(split(s), split(s')) \end{aligned} \quad (21)$$

## Flatten a cotree

Let  $T = \nu coTree(X)$  (see [Trees](#)). The functions  $flatten : T \rightarrow X^\infty$  and  $flattenL : T^\infty \rightarrow X^\infty$  satisfy the equations

$$split(flatten(t)) = (root(t), flattenL(subtrees(t))) \quad (1)$$

$$split(ts) = \epsilon \Rightarrow split(flattenL(ts)) = \epsilon \quad (2)$$

$$\begin{aligned} split(ts) &= (u, us) \\ \Rightarrow split(flattenL(ts)) &= (root(u), flattenL(conc(subtrees(u), us))) \end{aligned} \quad (3)$$

where  $conc : T^\infty \times T^\infty \rightarrow T^\infty$  is defined as in chapter 12.

$$(flatten, flattenL)^* = [flatten, flattenL] : T + T^\infty \rightarrow X^\infty$$

## Mirror a cobintree (see [78, 127])

Let  $T = \nu coBintree(X)_{btree}$ . The function  $mirror : T \rightarrow T$  satisfies the equations

$$split(t) = \epsilon \Rightarrow split(mirror(t)) = \epsilon \quad (1)$$

$$split(t) = (x, u, u') \Rightarrow split(mirror(t)) = (x, mirror(u'), mirror(u)) \quad (2)$$

Since  $T$  is a final algebra, properties of  $mirror^T$  like  $mirror^T \circ mirror^T = id_T$  are shown by algebraic coinduction (see, e.g., [127]).

## A language acceptor

$$\text{esum}, \text{osum} : 1 \rightarrow 2^{\mathbb{Z}^*}$$

$$\delta(\text{esum}) = \lambda x. \text{if even}(x) \text{ then esum else osum}$$

$$\beta(\text{esum}) = 1$$

$$\delta(\text{osum}) = \lambda x. \text{if even}(x) \text{ then osum else esum}$$

$$\beta(\text{osum}) = 0$$

$$\text{esum} = \text{obj}\{\delta. \lambda x. \text{if even}(x) \text{ then esum else osum}, \beta. 1\}$$

$$\text{osum} = \text{obj}\{\delta. \lambda x. \text{if even}(x) \text{ then osum else esum}, \beta. 0\}$$

$$\text{eosum} = [\text{esum}, \text{osum}] : 1 + 1 \rightarrow 2^{\mathbb{Z}^*}$$

$$\delta(\text{eosum}(\iota_1)) = \lambda x. \text{if even}(x) \text{ then eosum}(\iota_1) \text{ else eosum}(\iota_2)$$

$$\delta(\text{eosum}(\iota_2)) = \lambda x. \text{if even}(x) \text{ then eosum}(\iota_2) \text{ else eosum}(\iota_1)$$

$$\beta(\text{eosum}(\iota_1)) = 1$$

$$\beta(\text{eosum}(\iota_2)) = 0$$

$$\begin{aligned} \text{eosum} = \text{obj}\{ & \delta. \text{eosum} \circ \text{case}\{\iota_1. \text{if even then } \iota_1 \circ \lambda x. \epsilon \text{ else } \iota_2 \circ \lambda x. \epsilon \\ & \iota_2. \text{if even then } \iota_2 \circ \lambda x. \epsilon \text{ else } \iota_1 \circ \lambda x. \epsilon\}, \\ & \beta. \text{case}\{\iota_1. 1, \iota_2. 0\} \} \end{aligned} \tag{8}$$

## Sample biinductively defined functions

Functions into the final  $Stream(X)$ -algebra  $InfSeq(X)$  (see sample algebra 5)

Let  $X = \mathbb{N}$  and  $fibs : 1 \rightarrow state$  be a variable for the stream of Fibonacci numbers with the equations

$$\begin{aligned} head(fibs(z)) &= 0, \\ head(tail(fibs(z))) &= 1, \\ tail(tail(fibs(z))) &= add(fibs, tail(fibs)) \end{aligned}$$

(see [1], section 3.4; [37], Example 2.10). They entail the equations

$$\begin{aligned} head(c(\iota_1(z))) &= 0, \\ head(c(\iota_2(z))) &= 1, \\ tail(c(\iota_1(z))) &= c(\iota_2(z)), \\ tail(c(\iota_2(z))) &= add(c(\iota_1(z)), c(\iota_2(z))) \end{aligned}$$

for  $c =^{\mathcal{A}} [fibs, tail \circ fibs] : 1 + 1 \rightarrow state$ . An  $\mathcal{A}$ -equivalent coinductive definition of  $c$  reads as follows:

$$\begin{aligned} fibs &= obj\{head.zero, tail.fibs'\} \\ fibs' &= obj\{head.one, tail.add \circ \langle fibs, fibs' \rangle\} \end{aligned} \tag{1}$$

Let  $X = \mathbb{R}$ .

$$\begin{aligned}
 \text{mult} : \mathbb{R}^N \times \mathbb{R}^N &\rightarrow \mathbb{R}^N && (\text{shuffle product; see [148]}) \\
 \text{head}(\text{mult}(s, s')) &= \text{head}(s) * \text{head}(s') \\
 \text{tail}(\text{mult}(s, s')) &= \text{add}(\text{mult}(\text{tail}(s), s'), \text{mult}(s, \text{tail}(s'))) \\
 \text{mult} &= \text{obj}\{\text{head}.(*) \circ (\text{head} \times \text{head}), \\
 &\quad \text{tail.add} \circ \langle \text{mult} \circ (\text{tail} \times \text{id}), \text{mult} \circ (\text{id} \times \text{tail}) \rangle\} \tag{3}
 \end{aligned}$$

$$\begin{aligned}
 \text{conv} : \mathbb{R}^N \times \mathbb{R}^N &\rightarrow \mathbb{R}^N && (\text{convolution product; see [146, 148]; [26], Example3.1}) \\
 \text{head}(\text{conv}(s, s')) &= \text{head}(s) * \text{head}(s') \\
 \text{tail}(\text{conv}(s, s')) &= \text{add}(\text{conv}(\text{tail}(s), s'), \text{conv}(\text{appzeros}(\text{head}(s)), \text{tail}(s'))) \\
 \text{conv} &= \text{obj}\{\text{head}.(*) \circ (\text{head} \times \text{head}), \\
 &\quad \text{tail.add} \circ \langle \text{conv} \circ (\text{tail} \times \text{id}), \\
 &\quad \text{conv} \circ ((\text{appzeros} \circ \text{head}) \times \text{tail}) \rangle\} \tag{4}
 \end{aligned}$$

Remember that  $f_1 \times \cdots \times f_n$  stands for  $\langle f_i \circ \pi_1, \dots, f_n \circ \pi_n \rangle$ .

Coinductive definitions of  $\text{add}$  and  $\text{appzeros}$  are given above. May they be used here ???

## Biinductive definition of regular operators

Let  $(S, \mathcal{I}, C) = \text{Reg}(X)$ .

We transform the inductive definition  $E$  of the Brzozowski automaton given above into a biinductive definition  $E'$  of  $C$  on the final  $\text{Acc}(X)$ -algebra  $\text{Pow}(X)$  (see [sample algebra 20](#)).  $E$  uses concrete functions on  $T_{\text{Reg}(X)}$ , while  $E'$  must consist of uninterpreted equations of the form  $d_s \circ c = f_c \circ d_e$  with  $f_c \in \text{cl}(\text{Reg}(X))$  (see Theorem [RECFUN3](#)).

$$\text{eval} = \text{eval}^{\text{Pow}(X)} : \mathbb{D}_{\text{Acc}(X), \text{Bro}(X)}(\mathcal{P}(X^*)) \rightarrow T_{\text{Bro}(X)}(\mathcal{P}(X^*))$$

of  $\text{Acc}(X)$ -derivatives of  $\text{Bro}(X)$ -terms: For all  $t, u \in T_{\text{Reg}(X)}(\mathcal{P}(X^*))$ ,  $B \subseteq X$  and  $L \subseteq X^*$ ,

$$\begin{aligned} \text{eval}(\pi_x(\delta(L))) &= \delta^{\text{Pow}(X)}(L)(x), \\ \text{eval}(\pi_x(\delta(\text{par}(t, u)))) &= \text{par}(\text{eval}(\pi_x(\delta(t))), \text{eval}(\pi_x(\delta(u)))) \mid x \in X\}, \\ \text{eval}(\pi_x(\delta(\text{seq}(t, u)))) &= \text{if } \text{eval}(\beta(t)) = 1 \\ &\quad \text{then } \text{par}(\text{seq}(\text{eval}(\pi_x(\delta(t))), u), \text{eval}(\pi_x(\delta(u)))) \\ &\quad \text{else } \text{seq}(\text{eval}(\pi_x(\delta(t))), u), \\ \text{eval}(\pi_x(\delta(\text{iter}(t)))) &= \text{seq}(\text{eval}(\pi_x(\delta(t))), \text{iter}(t)), \\ \text{eval}(\pi_x(\delta(\text{eps}))) &= \text{eps}, \\ \text{eval}(\pi_x(\delta(\text{base}(B)))) &= \text{if } x \in B \text{ then } \text{eps} \text{ else } \text{base}(\emptyset), \\ \text{eval}(\beta(L)) &= \beta^{\text{Pow}(X)}(L), \end{aligned}$$

$$\begin{aligned}
 eval(\beta(par(t, u))) &= max(eval(\beta(t)), eval(\beta(u))), \\
 eval(\beta(seq(t, u))) &= eval(\beta(t)) * eval(\beta(u)), \\
 eval(\beta(iter(t))) &= 1, \\
 eval(\beta(eps)) &= 1, \\
 eval(\beta(base(B))) &= \text{if } B = 1 \text{ then } 1 \text{ else } 0.
 \end{aligned}$$

By Theorem RECFUN3, equations (1)-(8) (together with the equations for *add* and *appzeros*) have unique solutions in the final  $Stream(X)$ -algebra  $InfSeq(X)$  and the final  $Acc(X)$ -algebra  $Pow(X)$ , respectively (see chapters 11 and 12), and thus provide biinductive definitions of the functions colored red.

Let  $E = \{(5), (6), (7), (8)\}$ ,  $E'$  be the inductive definition of the Brzozowski automaton (see Sample inductively defined functions),  $\mathcal{A} = Pow(X)$  and  $\mathcal{A}' = Lang(X)$ . Then the assumptions of Corollary BIIND hold true. Hence

$$fold^{Lang(X)} = unfold^{Bro(X)} : T_{Reg(X)} \rightarrow \mathcal{P}(X^*).$$

## From constructors to destructors

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature,  $C_s = \{c \in C \mid \text{trg}(c) = s\}$ ,

$$\begin{aligned} D &= \{d_s : s \rightarrow \coprod_{c:e \rightarrow s \in C_s} e \mid s \in S, |C_s| > 1\} \\ &\cup \{d_s : s \rightarrow e \mid s \in S, C_s = \{c : e \rightarrow s\}\}, \\ co\Sigma &= (S, \mathcal{I}, D), \end{aligned}$$

$\mathcal{A}$  be an initial  $\Sigma$ -algebra with carrier  $A$  and  $B = \bigcup \mathcal{I}$ .

By Lambek's Lemma (1), the initial  $H_\Sigma$ -algebra

$$\alpha = \{\alpha_s : H_\Sigma(A)_s \xrightarrow{[c^\mathcal{A}]_{c:e \rightarrow s \in C}} A_s \mid s \in S\}$$

is iso (see  $\Sigma$ -functors). Consequently,

$$\{\alpha_s^{-1} : A_s \rightarrow H_\Sigma(A)_s \mid s \in S\}$$

is an  $H_\Sigma$ -coalgebra, which corresponds to the  $co\Sigma$ -algebra  $\mathcal{B}$  that is defined as follows:

For all  $s \in S$ ,  $\mathcal{B}(s) = A_s$  and  $d_s^\mathcal{B} = \alpha_s^{-1}$ . Hence for all  $c : e \rightarrow s \in C$ ,

$$d_s^\mathcal{B} \circ c^\mathcal{A} = \alpha_s^{-1} \circ c^\mathcal{A} \stackrel{(1)}{=} \alpha_s^{-1} \circ [c^\mathcal{A}]_{c:e \rightarrow s \in C} \circ \iota_c = \alpha_s^{-1} \circ \alpha_s \circ \iota_c = \iota_c.$$

Since  $co\Sigma$  is destructive, Theorem COFREE implies that  $DT_{co\Sigma}$  is final in  $Alg_{co\Sigma}$ .

$CT_\Sigma$  and, analogously,  $T_\Sigma$  are  $co\Sigma$ -algebras:

For all  $c : e \rightarrow s \in C$  and  $t \in CT_{\Sigma,e}$ ,

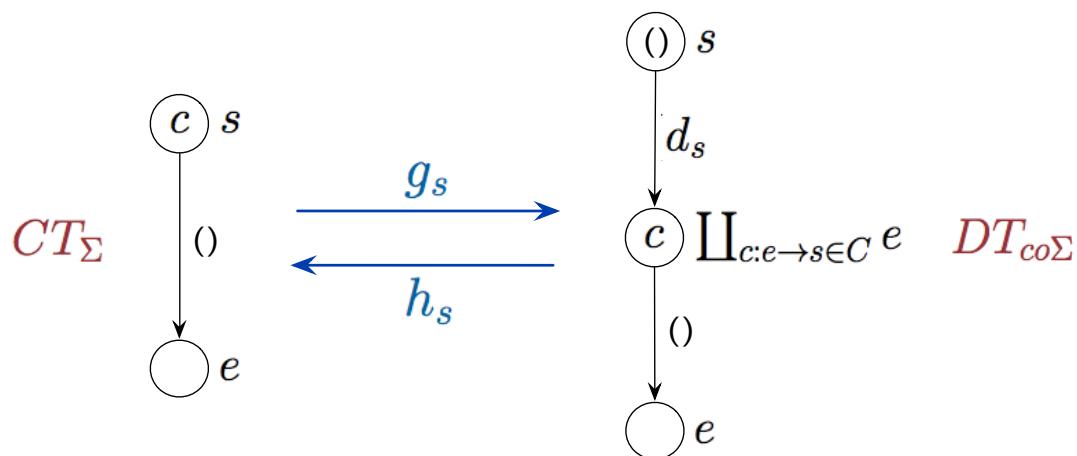
$$d_s^{CT_\Sigma}(c(t)) =_{def} c(t) \in \coprod_{c:e \rightarrow s \in C} CT_{\Sigma,e}.$$

$DT_{co\Sigma}$  and, analogously,  $coT_{co\Sigma}$  are  $\Sigma$ -algebras:

For all  $c : e \rightarrow s \in C$  and  $t \in DT_{co\Sigma,e}$ ,

$$c^{DT_{co\Sigma}}(t) =_{def} ()\{d_s \rightarrow c(t)\} \in DT_{co\Sigma,s}.$$

Note that, on the right-hand side of these equations,  $c$  is not a constructor, but a sum index.



The values of  $S$ -sorted functions  $g : CT_{\Sigma} \rightarrow DT_{co\Sigma}$  and  $h : DT_{co\Sigma} \rightarrow CT_{\Sigma}$  are defined inductively on  $(D \cup B)^*$  as follows:

For all  $c : e \rightarrow s \in C$ ,  $t \in CT_{\Sigma,e}$  and  $t' \in DT_{co\Sigma,e}$ ,

$$\begin{aligned} g_s(c(t)) &= ()\{d_s \rightarrow c(g_e(t))\}, \\ h_s(()\{d_s \rightarrow c(t')\}) &= c(h_e(t')). \end{aligned}$$

Bijective  $S$ -sorted functions  $g : T_{\Sigma} \rightarrow coT_{co\Sigma}$  and  $h : coT_{co\Sigma} \rightarrow T_{\Sigma}$  are defined analogously.

A simple proof by induction on  $(D \cup B)^*$  shows that  $g$  and  $h$  are inverse to each other.

Moreover,  $g$  is  $co\Sigma$ -homomorphic and  $h$  is  $\Sigma$ -homomorphic:

For all  $c : e \rightarrow s \in C$ ,  $t \in CT_{\Sigma,e}$  and  $t' \in DT_{co\Sigma,e}$ ,

$$\begin{aligned} g_{\coprod_{c:e \rightarrow s \in C} e}(d_s^{CT_{\Sigma}}(c(t))) &= g_{\coprod_{c:e \rightarrow s \in C} e}(c(t)) = g_{\coprod_{c:e \rightarrow s \in C} e}(\iota_c(t)) = \iota_c(g_e(t)) = c(g_e(t)) \\ &= d_s^{DT_{co\Sigma}}(()\{d_s \rightarrow c(g_e(t))\}) = d_s^{DT_{co\Sigma}}(g_s(c(t))), \\ h_s(c^{DT_{co\Sigma}}(t')) &= h_s(()\{d_s \rightarrow c(t')\}) = c(h_e(t')) = c^{CT_{\Sigma}}(h_e(t')). \end{aligned}$$

Since  $g$  is  $co\Sigma$ -homomorphic and  $g \circ h = id$ ,  $g \circ h$  and thus  $g$  are epi in  $Alg_{co\Sigma}$ . Hence by Lemma EMH (1),  $h$  is  $co\Sigma$ -homomorphic.

Since  $h$  is  $\Sigma$ -homomorphic and  $g \circ h = id$ ,  $g \circ h$  and thus  $h$  are mono in  $Alg_\Sigma$ . Hence by Lemma EMH (2),  $g$  is  $\Sigma$ -homomorphic.  $\square$

Therefore,  $CT_\Sigma$  and  $DT_{co\Sigma}$  and, analogously,  $T_\Sigma$  and  $coT_{co\Sigma}$  are both  $\Sigma$ - and  $co\Sigma$ -isomorphic. Consequently,  **$CT_\Sigma$  is final in  $Alg_{co\Sigma}$**  and  **$coT_{co\Sigma}$  is initial in  $Alg_\Sigma$** .

Given a  $co\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$ , the above definition of the bijection  $h : DT_{co\Sigma} \rightarrow CT_\Sigma$  implies that the  $S$ -components of  $unfold'^{\mathcal{A}} =_{def} h \circ unfold^{\mathcal{A}} : A \rightarrow CT_\Sigma$  (see State unfolding) are defined as follows: For all  $c : e \rightarrow s \in C$ ,  $a \in A_s$  and  $b \in A_e$ ,

$$d_s^{\mathcal{A}}(a) = \iota_c(b) \text{ implies } unfold'_s(a) = c(unfold'_e(b)).$$

*Proof.* Let  $d_s^{\mathcal{A}}(a) = \iota_c(b)$ . Then

$$unfold^{\mathcal{A}}(a) = ()\{d_s \rightarrow unfold^{\mathcal{A}}(\iota_c(b))\} = ()\{d_s \rightarrow c(unfold^{\mathcal{A}}(b))\}. \quad (2)$$

Hence

$$\begin{aligned} unfold'^{\mathcal{A}}(a) &= h(unfold^{\mathcal{A}}(a)) \stackrel{(1)}{=} h(()\{d_s \rightarrow c(unfold^{\mathcal{A}}(b))\}) = c(h(unfold^{\mathcal{A}}(b))) \\ &= c(unfold'^{\mathcal{A}}(b)). \end{aligned} \quad \square$$

## From destructors to constructors

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature,  $D_s = \{d \in D \mid \text{src}(d) = s\}$ ,

$$\begin{aligned} C &= \{c_s : \prod_{d:s \rightarrow e \in D_s} e \rightarrow s \mid d \in C_s, s \in S\} \\ &\quad \cup \{c_s : e \rightarrow s \mid s \in S, D_s = \{d : s \rightarrow e\}\}, \\ co\Sigma &= (S, \mathcal{I}, C), \end{aligned}$$

$\mathcal{A}$  be a final  $\Sigma$ -algebra with carrier  $A$  and  $B = \bigcup \mathcal{I}$ .

By Lambek's Lemma (2), the final  $H_\Sigma$ -coalgebra

$$\alpha = \{\alpha_s : A_s \xrightarrow{\langle d^A \rangle_{d:s \rightarrow e \in D}} H_\Sigma(A)_s \mid s \in S\}$$

is iso (see  $\Sigma$ -functors). Consequently,

$$\{\alpha_s^{-1} : H_\Sigma(A)_s \rightarrow A_s \mid s \in S\}$$

is an  $H_\Sigma$ -algebra, which corresponds to the  $co\Sigma$ -algebra  $\mathcal{B}$  that is defined as follows:

For all  $s \in S$ ,  $\mathcal{B}(s) = A_s$  and  $c_s^\mathcal{B} = \alpha_s^{-1}$ . Hence for all  $d : s \rightarrow e \in D$ ,

$$d^A \circ c_s^\mathcal{B} = d^A \circ \alpha_s^{-1} \stackrel{(2)}{=} \pi_d \circ \langle d^A \rangle_{d:s \rightarrow e \in D} \circ \alpha_s^{-1} = \pi_d \circ \alpha_s \circ \alpha_s^{-1} = \pi_d.$$

Since  $co\Sigma$  is constructive, Theorem FREE implies that  $T_{co\Sigma}$  is initial in  $Alg_{co\Sigma}$ .

$DT_\Sigma$  and, analogously,  $coT_\Sigma$  are  $co\Sigma$ -algebras:

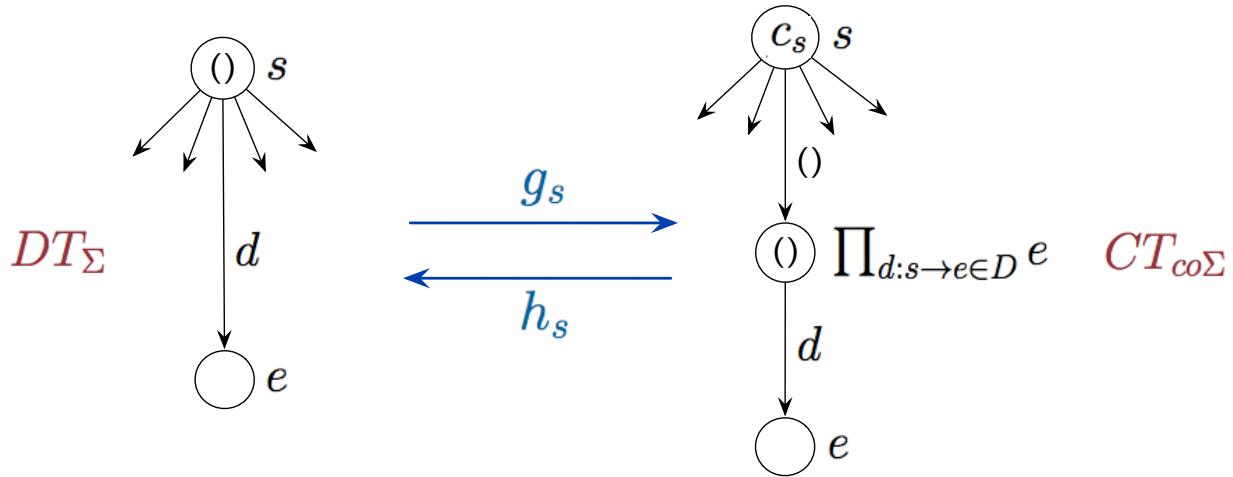
For all  $s \in S$  and  $t = (t_d)_{d:s \rightarrow e \in D} \in \prod_{d:s \rightarrow e \in D} DT_{\Sigma,e}$ ,

$$c_s^{DT_\Sigma}(t) \underset{def}{=} ()\{d \rightarrow t_d \mid d : s \rightarrow e \in D\} \in DT_{\Sigma,s}.$$

$CT_{co\Sigma}$  and, analogously,  $T_{co\Sigma}$  are  $\Sigma$ -algebras:

For all  $d : s \rightarrow e \in D$  and  $(t_d)_{d:s \rightarrow e \in D} \in \prod_{d:s \rightarrow e \in D} CT_{co\Sigma,e}$ ,

$$d^{CT_{co\Sigma}}((())\{d \rightarrow t_d \mid d : s \rightarrow e \in D\}) \underset{def}{=} t_d \in CT_{co\Sigma,e}.$$



The values of  $S$ -sorted functions  $g : DT_\Sigma \rightarrow CT_{co\Sigma}$  and  $h : CT_{co\Sigma} \rightarrow DT_\Sigma$  are defined inductively on  $(D \cup B)^*$  as follows:

For all  $s \in S$ ,  $t = ()\{d \rightarrow t_d \mid d : s \rightarrow e \in D\} \in DT_{\Sigma,s}$  and  $t' = c_s((())\{d \rightarrow t_d\} \mid d : s \rightarrow e \in D) \in CT_{co\Sigma,s}$ ,

$$\begin{aligned} g_s(t) &= c_s((())\{d \rightarrow g_e(t_d) \mid d : s \rightarrow e \in D\}), \\ h_s(t') &= ()\{d \rightarrow h_e(t_d) \mid d : s \rightarrow e \in D\}. \end{aligned}$$

Bijective  $S$ -sorted functions  $g : coT_\Sigma \rightarrow T_{co\Sigma}$  and  $h : T_{co\Sigma} \rightarrow coT_\Sigma$  are defined analogously.

A simple proof by induction on  $(D \cup B)^*$  shows that  $g$  and  $h$  are inverse to each other.

Moreover,  $g$  is  $co\Sigma$ -homomorphic and  $h$  is  $\Sigma$ -homomorphic:

For all  $s \in S$ ,  $t = ()\{d \rightarrow t_d \mid d : s \rightarrow e \in D\} \in DT_{\Sigma,s}$  and  $t' = c_s(()\{d \rightarrow t_d\} \mid d : s \rightarrow e \in D) \in CT_{co\Sigma,s}$ ,

$$\begin{aligned} g_s(c_s^{DT_\Sigma}(t)) &= g_s(()\{d \rightarrow t_d \mid d : s \rightarrow e \in D\}) \\ &= c_s(()\{d \rightarrow g_e(t_d) \mid d : s \rightarrow e \in D\}) = c_s^{CT_{co\Sigma}}(()\{d \rightarrow g_e(t_d) \mid d : s \rightarrow e \in D\}) \\ &= c_s^{CT_{co\Sigma}}(g_{\prod_{d:s \rightarrow e \in D}}(()\{d \rightarrow t_d \mid d : s \rightarrow e \in D\})) = c_s^{CT_{co\Sigma}}(g_{\prod_{d:s \rightarrow e \in D}}(t)), \\ h_e(d^{CT_{co\Sigma}}(t')) &= h_e(t_d) = d^{DT_\Sigma}(()\{d \rightarrow h_e(t_d) \mid d : s \rightarrow e \in D\}) = d^{DT_\Sigma}(h_s(t')). \end{aligned}$$

Since  $g$  is  $co\Sigma$ -homomorphic and  $g \circ h = id$ ,  $g \circ h$  and thus  $g$  are epi in  $Alg_{co\Sigma}$ . Hence by Lemma EMH (1),  $h$  is  $co\Sigma$ -homomorphic.

Since  $h$  is  $\Sigma$ -homomorphic and  $g \circ h = id$ ,  $g \circ h$  and thus  $h$  are mono in  $Alg_\Sigma$ . Hence by Lemma EMH (2),  $g$  is  $\Sigma$ -homomorphic.  $\square$

Therefore,  $DT_\Sigma$  and  $CT_{co\Sigma}$  and, analogously,  $coT_\Sigma$  and  $T_{co\Sigma}$  are both  $\Sigma$ - and  $co\Sigma$ -isomorphic. Consequently,  **$CT_{co\Sigma}$  is final in  $Alg_\Sigma$**  and  **$coT_\Sigma$  is initial in  $Alg_{co\Sigma}$** .

Given a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$ , the above definition of the bijection  $g : coT_\Sigma \rightarrow T_{co\Sigma}$  implies that the  $S$ -components of  $unfold'^{\mathcal{A}} = g \circ unfold^{\mathcal{A}} : A \rightarrow CT_{co\Sigma}$  (see **State unfolding**) are defined as follows: For all  $s \in S$  and  $a \in A_s$ ,

$$unfold'_s(a) = c_s((d \rightarrow unfold'_e(d^{\mathcal{A}}(a)) \mid d : s \rightarrow e \in D)).$$

*Proof.*

$$\begin{aligned} unfold'^{\mathcal{A}}(a) &= g(unfold^{\mathcal{A}}(a)) = g((d \rightarrow unfold^{\mathcal{A}}(d^{\mathcal{A}}(a)) \mid d : s \rightarrow e \in D)) \\ &= c_s((d \rightarrow g(unfold^{\mathcal{A}}(d^{\mathcal{A}}(a)))) \mid d : s \rightarrow e \in D) \\ &= c_s((d \rightarrow unfold'^{\mathcal{A}}(d^{\mathcal{A}}(a))) \mid d : s \rightarrow e \in D). \end{aligned} \quad \square$$

## Continuous algebras

**$Poset$**  denotes the category of partially ordered sets with a least element as objects and strict and monotone functions as morphisms.

$CPO$  denotes the category of  $\omega$ -CPOs as objects and strict and  $\omega$ -continuous functions as morphisms (see Orders, lattices and fixpoints).

$Poset^S$  denotes the subcategory of  $Set^S$  that consists of all  $S$ -tuples of objects or morphisms of  $Poset$ . For all  $A \in Poset^S$  and  $s \in S$ ,  $\perp_s^A$  denotes the least element of  $A_s$ .

$CPO^S$  denotes the subcategory of  $Set^S$  that consists of all  $S$ -tuples of objects or morphisms of  $CPO$ .

Every  $A \in Set^{\mathcal{T}_p(S, \mathcal{I})}$  is lifted to an object of  $Poset^{\mathcal{T}_p(S, \mathcal{I})}$  and  $CPO^{\mathcal{T}_p(S, \mathcal{I})}$  as follows:

- $A_1 = 1$ .
- For all  $I \in \mathcal{I}$  and  $(e_i)_{i \in I} \in \mathcal{T}_p(S, \mathcal{I})^I$ ,

$$A_{\prod_{i \in I} e_i} = \prod_{i \in I} A_{e_i} \quad \text{and} \quad A_{\coprod_{i \in I} e_i} = \coprod_{i \in I} A_{e_i} \cup \{\perp_{\coprod_{i \in I} e_i}\}.$$

The partial orders, least elements and suprema of  $\omega$ -chains are defined as follows:

- For all  $I \in \mathcal{I}$ ,  $(e_i)_{i \in I} \in \mathcal{T}_p(S, \mathcal{I})^I$ ,  $a, b \in A_{\prod_{i \in I} e_i}$  and  $\omega$ -chains  $C$  of  $A_{\prod_{i \in I} e_i}$  and  $i \in I$ ,

$$a \leq_{\prod_{i \in I} e_i}^A b \Leftrightarrow \forall i \in \mathbb{N} : \pi_i(a) \leq_{e_i} \pi_i(b),$$

$$\begin{aligned}\pi_i(\perp_{\prod_{i \in I} e_i}^A) &= \perp_{e_i}^A, \\ \pi_i(\bigsqcup C) &= \bigsqcup \{\pi_i(a) \mid a \in C\}.\end{aligned}$$

- For all  $I \in \mathcal{I}$ ,  $(e_i)_{i \in I} \in \mathcal{T}_p(S, \mathcal{I})^I$ ,  $a, b \in A_{\coprod_{i \in I} e_i}$  and  $\omega$ -chains  $C$  of  $A_{\coprod_{i \in I} e_i}$ ,

$$\begin{aligned}a \leq_{\coprod_{i \in I} e_i}^A b &\Leftrightarrow a = \perp_{\coprod_{i \in I} e_i} \vee \\ &\exists i \in I, a', b' \in A_{e_i} : a' \leq_{e_i}^A b' \wedge \iota_i(a') = a \wedge \iota_i(b') = b. \\ \bigsqcup C &= \begin{cases} \perp_{\coprod_{i \in I} e_i} & \text{if } \forall C = \{\perp_{\coprod_{i \in I} e_i}\}, \\ \bigsqcup \{a \in A_{e_i} \mid \iota_i(a) \in C, i \in I\} & \text{otherwise.} \end{cases}\end{aligned}$$

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature.

$PAlg_\Sigma$  denotes the category of all  $\Sigma$ -algebras with carrier  $A \in Poset^S$ , monotonic operations (w.r.t. the above lifting of  $A$  to an object of  $Poset^{\mathcal{T}_p(S, \mathcal{I})}$ ) and all  $\Sigma$ -homomorphisms in  $Mor(Poset^S)$ . The objects of  $PAlg_\Sigma$  are called **monotone  $\Sigma$ -algebras**.

$CAlg_\Sigma$  denotes the category of all  $\Sigma$ -algebras with carrier  $A \in CPO^S$  and  $\omega$ -continuous operations (w.r.t. the above lifting of  $A$  to an object of  $CPO^{\mathcal{T}_p(S, \mathcal{I})}$ ) and all  $\Sigma$ -homomorphisms in  $Mor(CPO^S)$ . The objects of  $CAlg_\Sigma$  are called  **$\omega$ -continuous  $\Sigma$ -algebras**.

## Proposition MON2

Let  $\mathcal{A}$  be a monotone  $\Sigma$ -algebra with carrier  $A$  such that for all  $s \in S$ ,  $A_s$  is chain-finite (see chapter 3). Then  $\mathcal{A}$  is  $\omega$ -continuous.

*Proof.* Since for all  $e \in \mathcal{T}_p(S, \mathcal{I})$ ,  $A_e$  is chain-finite, Proposition MON (4) implies that all operations of  $\mathcal{A}$  are  $\omega$ -continuous.  $\square$

## Both terms and flowcharts form a CPO

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature,  $B = \bigcup \mathcal{I}$  and  $V$  be an  $S$ -sorted set of “variables”.

The sets  $CT_{\Sigma}^{\perp}(V)$  and  $T_{\Sigma}^{\perp}(V)$  of (well-founded) **ordered  $\Sigma$ -terms over  $V$**  are defined the same as  $CT_{\Sigma}(V)$  and  $T_{\Sigma}(V)$ , respectively, except that (1), (3), (6) and (7) in section  $\Sigma$ -terms and -coterm are replaced as follows:

- For all  $s \in S$  and  $t \in M_s$ ,  $t \in V_s \cup \{\Omega\}$  (see **Miscellanea**) or there are  $c : e \rightarrow s \in C$  and  $u \in M_e$  such that  $t = c(u)$ .  $(1')$
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t \in M_e$ ,  $t = \Omega$  or there are  $i \in I$  and  $u \in M_{e_i}$  such that  $t = i(u)$ .  $(3')$
- For all  $s \in S$ ,  $V_s \cup \{\Omega\} \subseteq M_s$ .  $(6')$

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in M_{e_i}$ ,  $\Omega, i(t) \in M_e$ . (7')

Let  $V \in Set_b^S$  (see chapter 7) and for all  $s \in S$ , let  $V_s = \emptyset$ . Then the elements of  $\overline{CT}_{\Sigma}^{\perp} =_{def} CT_{\Sigma}^{\perp}(V)$  und  $\overline{T}_{\Sigma}^{\perp} =_{def} T_{\Sigma}^{\perp}(V)$  are called **ground ordered  $\Sigma$ -terms**.

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature,  $B = \bigcup \mathcal{I}$  and  $V$  be an  $S$ -sorted set of “variables”.

The sets  $\overline{CT}_{\Sigma}^{\perp}(V)$  and  $\overline{T}_{\Sigma}^{\perp}(V)$  of (well-founded) **ordered  $\Sigma$ -flowcharts over  $V$**  are defined the same as  $\overline{CT}_{\Sigma}(V)$  and  $\overline{T}_{\Sigma}(V)$ , respectively, except that (1), (2), (5) and (6) in section  **$\Sigma$ -flowcharts** are replaced as follows:

- For all  $s \in S$  and  $t \in M_s$ ,  $t \in V_s \cup \{\Omega\}$  or there are  $d : s \rightarrow e \in F$  and  $u \in M_e$  such that  $t = d(u)$ . (1')
- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t \in M_e$ ,  $t = \Omega$  or there are  $i \in I$  and  $u \in M_{e_i}$  such that  $t = i(u)$ . (2')
- For all  $s \in S$ ,  $V_s \cup \{\Omega\} \subseteq M_s$ . (5')
- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in M_{e_i}$ ,  $\Omega, i(t) \in M_e$ . (6')

$T_\Sigma^\perp(V), \overline{T_\Sigma^\perp}(V) \in Poset^S$  and  $CT_\Sigma^\perp(V), \overline{CT_\Sigma^\perp}(V) \in CPO^S$ .

*Proof.*

For all  $s \in S$ ,  $\Omega$  is the least element of  $T \in \{T_\Sigma^\perp(V)_s, \overline{T_\Sigma^\perp}(V)_s, CT_\Sigma^\perp(V)_s, \overline{CT_\Sigma^\perp}(V)_s\}$  and for all  $t, t' \in T$ ,

$$t \leq_s t' \Leftrightarrow_{def} \forall w \in def(t) : t(w) = t'(w).$$

Every  $\omega$ -chain  $T \subseteq CT_\Sigma^\perp(V)_s$  or  $T \subseteq \overline{CT_\Sigma^\perp}(V)_s$  has a supremum: For all  $w \in B^*$ ,

$$(\bigsqcup T)(w) = \begin{cases} t(w) & \text{if } \exists t \in T : w \in def(t), \\ \perp & \text{otherwise.} \end{cases} \quad \square$$

$T_\Sigma^\perp(V) \in PAlg_\Sigma$  and  $CT_\Sigma^\perp(V) \in CAlg_\Sigma$ .

*Proof.* The arrows of  $\Sigma$  are interpreted in  $CT_\Sigma(V)^\perp$  as in  $CT_\Sigma(V)$ . The interpretations are  $\omega$ -continuous.

$T_\Sigma^\perp(V)$  is a monotone  $\Sigma$ -subalgebra of  $CT_\Sigma^\perp(V)$ .  $\square$

### Theorem PFREE (generalization of [55], Prop. 4.7)

$T_\Sigma^\perp(V)$  is free over  $V$  in  $PAlg_\Sigma$ . In particular,  $T_\Sigma^\perp$  is initial in  $PAlg_\Sigma$ .

*Proof.* Let  $\mathcal{A}$  be a monotone  $\Sigma$ -algebra with carrier  $A$  and  $g \in A^V$ .

$$\begin{array}{ccc}
 V & \xrightarrow{\text{inc}_V} & T_\Sigma^\perp(V) \\
 & \searrow g & \swarrow g^* \\
 & (3) &
 \end{array}$$

The **monotone term extension**  $g^* : T_\Sigma^\perp(V) \rightarrow A$  of  $g$  is the  $S$ -sorted function that is defined on  $T_\Sigma(V)$  the same as  $g^* : T_\Sigma(V) \rightarrow A$  (see [Term folding](#)). In addition,

- for all  $s \in S$ ,  $g_s^*(\Omega) = \perp_s^A$ .

$g^*$  is strict, monotone and  $\Sigma$ -homomorphic.

The uniqueness of  $g^*$  w.r.t. (3) can be shown analogously to the proof of Theorem [FREE](#). Hence we conclude that  $T_\Sigma^\perp(V)$  is free over  $V$  in  $PAlg_\Sigma$ .  $\square$

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature. For all  $n \in \mathbb{N}$ , the  $\mathcal{T}_p(S, \mathcal{I})$ -sorted function  $\underline{|}_n : CT_\Sigma^\perp(V) \rightarrow T_\Sigma^\perp(V)$  is defined inductively as follows:

For all  $t \in CT_\Sigma^\perp(V)$ ,  $x \in V \cup \{\Omega\}$ ,  $c : e \rightarrow s \in C$ ,  $u \in T_\Sigma^\perp(V)_e$ ,  $I \in \mathcal{I}$ ,  $i \in I$  and  $(t_i)_{i \in I} \in \bigtimes_{i \in I} T_\Sigma^\perp(V)_{e_i}$ ,

$$\begin{aligned} t|_0 &= \Omega, \\ x|_{n+1} &= x, \\ c(u)|_{n+1} &= c(u|_n), \\ i(t_i)|_{n+1} &= i(t_i|_n), \\ ()\{i \rightarrow t_i \mid i \in I\}|_{n+1} &= ()\{i \rightarrow t_i|_n \mid i \in I\}. \end{aligned}$$

Hence  $t = \bigsqcup_{n < \omega} t|_n$ .

Given a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$ , we extend the functional interpretation of well-founded  $\Sigma$ -terms to arbitrary ones: For all  $t \in CT_\Sigma^\perp(V)$ ,

$$t^{\mathcal{A}} \underset{\text{def}}{=} \bigsqcup_{n < \omega} (t|_n)^{\mathcal{A}}.$$

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature. For all  $n \in \mathbb{N}$ , the  $\mathcal{T}_p(S, \mathcal{I})$ -sorted function  $\underline{\phantom{x}}|_n : \overline{CT_\Sigma^\perp}(V) \rightarrow \overline{T_\Sigma}^\perp(V)$  is defined inductively as follows:

For all  $t \in \overline{CT_\Sigma^\perp}(V)$ ,  $x \in V \cup \{\Omega\}$ ,  $d : s \rightarrow e \in D$ ,  $u \in \overline{T_\Sigma}^\perp(V)_e$ ,  $I \in \mathcal{I}$ ,  $i \in I$  and  $(t_i)_{i \in I} \in \mathsf{X}_{i \in I} \overline{T_\Sigma}^\perp(V)_{e_i}$ ,

$$\begin{aligned} t|_0 &= \Omega, \\ x|_{n+1} &= x, \\ c(u)|_{n+1} &= c(u|_n), \\ i(t_i)|_{n+1} &= i(t_i|_n), \\ ()\{i \rightarrow t_i \mid i \in I\}|_{n+1} &= ()\{i \rightarrow t_i|_n \mid i \in I\}. \end{aligned}$$

Hence  $t = \bigsqcup_{n < \omega} t|_n$ .

Given a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$ , we extend the interpretation of well-founded  $\Sigma$ -flowcharts to arbitrary ones: For all  $t \in \overline{CT_\Sigma^\perp}(V)$ ,

$$t^{\mathcal{A}} =_{def} \bigsqcup_{n < \omega} (t|_n)^{\mathcal{A}}.$$

## $\omega$ -Completion Theorem

Let  $A \in CPO^S$ ,  $\Sigma$  be a constructive polynomial signature and  $f : T_\Sigma^\perp(V) \rightarrow A$  be strict and monotone. Then

$$\begin{aligned} f_\omega : CT_\Sigma^\perp(V) &\rightarrow A \\ t &\mapsto \bigsqcup\{f(t|_n) \mid n \in \mathbb{N}\} \end{aligned}$$

is strict and  $\omega$ -continuous.

Moreover, if  $A$  is the carrier of an  $\omega$ -continuous  $\Sigma$ -algebra and  $f$  is  $\Sigma$ -homomorphic, then  $f_\omega$  is  $\Sigma$ -homomorphic.

Let  $A \in CPO^S$ ,  $\Sigma$  be a destructive polynomial signature and  $f : \overline{T_\Sigma}^\perp(V) \rightarrow A$  be strict and monotone. Then

$$\begin{aligned} f_\omega : \overline{CT_\Sigma^\perp}(V) &\rightarrow A \\ t &\mapsto \bigsqcup\{f(t|_n) \mid n \in \mathbb{N}\} \end{aligned}$$

is strict and  $\omega$ -continuous.

*Proof.* See the proof of [55], Thm. 4.8. □

**Theorem CFREE** (generalization of [55], Cor. 4.9)

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature.  $CT_{\Sigma}^{\perp}(V)$  is free over  $V$  in  $CAlg_{\Sigma}$ . In particular,  $CT_{\Sigma}^{\perp}$  is initial in  $CAlg_{\Sigma}$ .

*Proof.*

For all  $c : e \rightarrow s \in C$ ,  $c^{CT_{\Sigma}^{\perp}(V)}$  is  $\omega$ -continuous:

Let  $T$  be an  $\omega$ -chain of  $CT_{\Sigma}^{\perp}(V)_e = \prod_{i \in I} CT_{\Sigma}^{\perp}(V)_{s_i}$ . Then

$$c^{CT_{\Sigma}^{\perp}(V)}(\bigsqcup T) = c(\bigsqcup T) = c(\bigsqcup \{t \mid t \in T\}) = \bigsqcup \{c(t) \mid t \in T\} = \bigsqcup \{c^{CT_{\Sigma}^{\perp}(V)}(t) \mid t \in T\}.$$

Let  $\mathcal{A}$  be an  $\omega$ -continuous  $\Sigma$ -algebra with carrier  $A$  and  $g \in A^V$ . Then  $\mathcal{A}$  is monotone and thus, by the initiality of  $T_{\Sigma}^{\perp}(V)$  in  $PAlg_{\Sigma}$  there is a unique strict and monotone  $\Sigma$ -homomorphism  $g^* : T_{\Sigma}^{\perp}(V) \rightarrow \mathcal{A}$ .

By the  $\omega$ -Completion Theorem,  $g_{\omega}^* : CT_{\Sigma}^{\perp}(V) \rightarrow \mathcal{A}$  is strict,  $\omega$ -continuous and  $\Sigma$ -homomorphic.

$$\begin{array}{ccc} V & \xrightarrow{inc_V} & CT_{\Sigma}^{\perp}(V) \\ & \searrow g & \swarrow g_{\omega}^* \\ & (4) & \\ & A & \end{array}$$

For the proof that there is at most one strict and  $\omega$ -continuous  $\Sigma$ -homomorphism from  $CT_\Sigma^\perp(V)$  to  $A$  satisfying (4), consult [55], Thm. 4.8, [21], Thm. 3.2, or [5], Prop. IV.2.

If for all  $s \in S$ ,  $V_s = \emptyset$ , then  $g_\omega^*$  no longer depends on  $g$  and thus agrees with the  $\omega$ -completion  $\text{fold}_\omega^A : CT_\Sigma^\perp \rightarrow A$  of the unique monotonic  $\Sigma$ -homomorphism  $\text{fold}^A : T_\Sigma^\perp \rightarrow A$  (see Theorem PFREE).  $\square$

We conclude that non-well-founded elements of  $CT_\Sigma$  can be regarded as suprema of  $\omega$ -chains of well-founded ones. Together with the initiality of  $T_\Sigma$  in  $Alg_\Sigma$  and the finality of  $CT_\Sigma$  in  $Alg_{co\Sigma}$ , Theorem CFREE entails the following corollary:

The final  $co\Sigma$ -algebra is a completion of the initial  $\Sigma$ -algebra (see [21], Thm. 3.2; [5], Prop. IV.2).

## Lemma SUBSTC

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature and  $V, V'$  be  $S$ -sorted sets of variables. For all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$ , substitutions  $g : V \rightarrow CT_\Sigma^\perp(V')$  and term valuations  $h : V' \rightarrow A$ ,

$$(h_\omega^* \circ g)^* = h_\omega^* \circ g^* : T_\Sigma(V) \rightarrow CT_\Sigma^\perp(V'). \quad (1)$$

*Proof.* By the  $\omega$ -Completion Theorem,  $h_\omega^*$  is  $\Sigma$ -homomorphic.

Hence by Lemma **SUBST**, (1) holds true. □

## Lemma SUBSTFC

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature and  $V, V'$  be  $S$ -sorted sets of variables. For all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$ , flowchart substitutions  $g : V \rightarrow \overline{CT}_\Sigma^\perp(V')$  and flowchart valuations  $h : V' \rightarrow Val^A$ ,

$$(h_\omega^+ \circ g)^+ = h_\omega^+ \circ g^* : (\overline{T}_\Sigma(V)_e \rightarrow B^{A_e})_{e \in T_p(S, \mathcal{I})}.$$

*Proof.* Let  $t \in \overline{T}_\Sigma(V)$ . We show

$$(h_\omega^+ \circ g)^+(t) = h_\omega^+(g^*(t)) \tag{2}$$

by induction on  $t$ .

*Case 1.*  $t \in V$ . Then  $(h_\omega^+ \circ g)^+(t) = h_\omega^+(g(t)) = h_\omega^+(g^*(t))$ .

*Case 2.*  $t = d(u)$  for some  $d : s \rightarrow e \in D$  and  $u \in \overline{T}_\Sigma(V)$ . Then

$$\begin{aligned} (h_\omega^+ \circ g)^+(t) &= (h_\omega^+ \circ g)^+(u) \circ d^{\mathcal{A}} \stackrel{ind.}{=} h_\omega^+(g^*(u)) \circ d^{\mathcal{A}} = (\bigsqcup_{n < \omega} h^+(g^*(u)|_n)) \circ d^{\mathcal{A}} \\ &= \bigsqcup_{n < \omega} (h^+(g^*(u)|_n) \circ d^{\mathcal{A}}) = \bigsqcup_{n < \omega} h^+(d(g^*(u)|_n)) = \bigsqcup_{n < \omega} h^+(d(g^*(u))|_{n+1}) \end{aligned}$$

$$= \bigsqcup_{n < \omega} h^+(d(g^*(u))|_n) = h_\omega^+(d(g^*(u))) = h_\omega^+(g^*(d(u))) = h_\omega^+(g^*(t)).$$

Case 3.  $t = i(u)$  for some  $i \in I$ ,  $I \in \mathcal{I}$ ,  $u \in \overline{T_\Sigma}(V)_e$  and  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then (2) follows analogously to Case 2 with  $i$  instead of  $d$  and  $\pi_i$  instead of  $d^\mathcal{A}$ .

Case 4.  $t = ()\{i \rightarrow t_i \mid i \in I\}$  for some  $I \in \mathcal{I}$ ,  $(t_i)_{i \in I} \in \bigtimes_{i \in I} \overline{T_\Sigma}(V)_{e_i}$  and  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ . Then

$$\begin{aligned} (h_\omega^+ \circ g)^+(t) &= (h_\omega^+ \circ g)^+((())\{i \rightarrow t_i \mid i \in I\}) = [(h_\omega^+ \circ g)^+(t_i)]_{i \in I} \stackrel{\text{ind. hyp.}}{=} [h_\omega^+(g^*(t_i))]_{i \in I} \\ &= [\bigsqcup_{n < \omega} h^+(g^*(t_i)|_n)]_{i \in I} = \bigsqcup_{n < \omega} [h^+(g^*(t_i)|_n)]_{i \in I} = \bigsqcup_{n < \omega} h^+((())\{i \rightarrow g^*(t_i)|_n \mid i \in I\}) \\ &= \bigsqcup_{n < \omega} h^+((())\{i \rightarrow g^*(t_i) \mid i \in I\}|_{n+1}) = \bigsqcup_{n < \omega} h^+(g^*(t)|_{n+1}) = \bigsqcup_{n < \omega} h^+(g^*(t)|_n) \\ &= h_\omega^+(g^*(t)). \end{aligned}$$

□

## Iterative equations

Given a signature  $\Sigma = (S, \mathcal{I}, F)$ , iterative equations are  $\Sigma$ -equations with a single variable on the left-hand side. They may represent circular  $\Sigma$ -terms or -flowcharts and can be solved in *algebraic* or *sorted theories* [31, 171, 172], which can be regarded as  $\Sigma$ -algebras from the subcategory  $Pow(\Sigma)$  or  $Sum(\Sigma)$  of  $\mathcal{K}(\Sigma)$  to  $Set$  (see chapter 9).

In the case of  $Pow(\Sigma)$ ,  $F$  consists of constructors with product source and  $Arr_\Sigma$  is restricted to projections and product extensions. In the case of  $Sum(\Sigma)$ ,  $F$  consists of destructors with sum target and  $Arr_\Sigma$  is restricted to injections and sum extensions. Hence for all  $Pow(\Sigma)$ -morphisms  $f : e \rightarrow e'$ ,  $e$  and  $e'$  are sorts or product types, while for all  $Sum(\Sigma)$ -morphisms  $f : e \rightarrow e'$ ,  $e$  and  $e'$  are sorts or sum types.

Accordingly, the algebraic theories  $Pow_A$  and  $Sum_A$  defined in [171] (Exs. 2.2, 2.3) and [172] (Exs. 2.4.2, 2.4.7) correspond to  $\Sigma$ -algebras  $\mathcal{A} : Pow(\Sigma) \rightarrow Set$  and  $\mathcal{B} : Sum(\Sigma) \rightarrow Set$  with carrier  $A$  and  $B$ , respectively. For all  $Pow(\Sigma)$ -morphisms  $f$  and  $Sum(\Sigma)$ -morphisms  $g$ ,  $f^{\mathcal{A}} : A^I \rightarrow A^V$  and  $g^{\mathcal{A}} : V \times A \rightarrow O \times A$  for sets  $I, V, O$  of variables, which actually represent product or sum indices, respectively.

## Term equations

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature, and  $V, I$  be finite  $S$ -sorted sets of “internal” and “input variables”, respectively. An  $S$ -sorted function

$$E : V \rightarrow T_\Sigma(I + V)$$

is called a **system of iterative  $\Sigma$ -equations** if  $\text{img}(E) \cap (I + V) = \emptyset$ .

Hence for all  $s \in S$  and  $x \in V_s$ ,  $E(x) = c(t)$  for some  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(I + V)_e$ .

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .

$f : A^I \rightarrow A^V$  solves  $E$  in  $\mathcal{A}$  if for all  $h \in A^I$ ,

$$[h, f(h)]^* \circ E = f(h).$$

Hence  $f$  solves  $E$  in  $\mathcal{A}$  iff  $f$  is a fixpoint of the following step function:

$$\begin{aligned} E^\mathcal{A} : (A^I \rightarrow A^V) &\rightarrow (A^I \rightarrow A^V) \\ f &\mapsto \lambda h.[h, f(h)]^* \circ E \end{aligned}$$

## The CPO approach for computing solutions of term equations

Let  $\mathcal{A}$  be  $\omega$ -continuous. According to the section [Continuous algebras](#), the partial orders, least elements and suprema of  $A$  can be lifted to  $A^V$  and then to  $A^I \rightarrow A^V$ , i.e.,  $A^V$  and  $A^I \rightarrow A^V$  are  $\omega$ -CPOs.

$E^{\mathcal{A}}$  is  $\omega$ -continuous. Hence by Kleene's Fixpoint Theorem (1),

$$(E^{\mathcal{A}})^{\infty} = \bigsqcup_{n<\omega} (E^{\mathcal{A}})^n(\perp) : A^I \rightarrow A^V$$

is the least fixpoint of  $E^{\mathcal{A}}$  where  $\perp : A^I \rightarrow A^V$  maps all functions of  $A^I$  to the least function of  $A^V$ , which maps all elements of  $V$  to the least element of  $A$ .

**Lemma TEQ** For all  $g : V \rightarrow CT_{\Sigma}(I)$ ,

$$[inc_I, g]^* \circ E = g \quad \Rightarrow \quad f_g \text{ solves } E \text{ in } \mathcal{A} \quad (1)$$

where  $f_g : A^I \rightarrow A^V$  maps  $h \in A^I$  to  $V \xrightarrow{g} CT_{\Sigma}(I) \xrightarrow{h_{\omega}^*} A$  (see Continuous algebras).

*Proof.* Suppose that

$$[inc_I, g]^* \circ E = g. \quad (2)$$

Then for all  $h \in A^I$ ,

$$\begin{aligned} [h, f_g(h)]^* \circ E &= [h, h_{\omega}^* \circ g]^* \circ E = [h^* \circ inc_I, h_{\omega}^* \circ g]^* \circ E = [h_{\omega}^* \circ inc_I, h_{\omega}^* \circ g]^* \circ E \\ &= (h_{\omega}^* \circ [inc_I, g])^* \circ E \stackrel{\text{Lemma SUBSTC}}{=} h_{\omega}^* \circ [inc_I, g]^* \circ E \stackrel{(2)}{=} h_{\omega}^* \circ g = f_g(h), \end{aligned}$$

i.e.,  $f_g$  solves  $E$  in  $\mathcal{A}$ . □

**Theorem SOLC** (generalization of [55], Thm. 5.2; [170], Thm. 6.15; [107], Satz 17)

Let  $E : V \rightarrow T_\Sigma(I + V)$  be a system of iterative  $\Sigma$ -equations. There is exactly one  $g : V \rightarrow CT_\Sigma(I)$  with (2).

*Proof.* Define the step function  $E_C : CT_\Sigma^\perp(I)^V \rightarrow CT_\Sigma^\perp(I)^V$  follows:

For all  $g : V \rightarrow CT_\Sigma^\perp(I)$ ,

$$E_C(g) = [inc_I, g]^* \circ E.$$

Since  $E_C$  is  $\omega$ -continuous, Kleene's Fixpoint Theorem (1) implies that

$$E_C^\infty = \bigsqcup_{n < \omega} E_C^n(\perp) : V \rightarrow CT_\Sigma^\perp(I)$$

is the least fixpoint of  $E_C$  where  $\perp : V \rightarrow CT_\Sigma^\perp(I)$  maps every  $x \in V$  to  $\Omega$ .

Hence it remains to show that every  $g : V \rightarrow CT_\Sigma(I)$  with (2) agrees with  $E_C^\infty$ .

So let  $B = \bigcup \mathcal{I}$  and  $g : V \rightarrow CT_\Sigma(I)$  satisfy (2). Since  $E_C^\infty$  is the least function that satisfies (2),

$$E_C^\infty \leq g. \tag{3}$$

Below we show that for all  $t \in T_\Sigma(I + V)$  and  $n \in \mathbb{N}$ ,

$$\text{def}([inc_I, g]^*(t)) \cap B^n \subseteq \text{def}([inc_I, E_C^{n+1}(\perp)]^*(t)), \quad (4)$$

in particular, for all  $x \in V$ ,

$$\text{def}(g(x)) \cap B^n \subseteq \text{def}(E_C^{n+1}(\perp)(x)). \quad (5)$$

(5) implies

$$\text{def}(g(x)) \subseteq \bigcup_{n < \omega} \text{def}(E_C^n(\perp)(x)) = \text{def}(\bigsqcup_{n < \omega} E_C^n(\perp)(x)) = \text{def}(E_C^\infty(x))$$

and thus  $g \leq E_C^\infty$ . Hence by (3),  $g = E_C^\infty$ .

*Proof of (4) by induction on n.*

Let  $t \in T_\Sigma(I + V)$ ,  $n \in \mathbb{N}$ ,  $\textcolor{red}{h} = [inc_I, g]$  and  $\textcolor{red}{h}_n = [inc_I, E_C^n(\perp)]$ .

*Case 1:*  $t = *$ . Then  $\text{def}([inc_I, g]^*(t)) \cap B^0 = 1$ , for all  $n > 0$ ,  $\text{def}([inc_I, g]^*(t)) \cap B^n = \emptyset$ , and for all  $n \in \mathbb{N}$ ,  $\text{def}([inc_I, E_C^{n+1}(\perp)]^*(t)) = 1$ . Hence (4) holds true.

*Case 2:*  $t \in V$  and  $E(t) = c(u)$  for some  $c : e \rightarrow s \in C$  and  $u \in T_\Sigma(I + V)_e$ . By (2),

$$h^*(t) = g(t) = h^*(E(t)) = h^*(c(u)) = c^A(h^*(u)) = c(h^*(u)), \quad (6)$$

$$\begin{aligned} h_{n+1}^*(t) &= E_C^{n+1}(\perp)(t) = E_C(E_C^n(\perp))(t) = h_n^*(E(t)) = h_n^*(c(u)) = c^A(h_n^*(u)) \\ &= c(h_n^*(u)). \end{aligned} \quad (7)$$

Case 2.1:  $n = 0$ . Then

$$\text{def}(h^*(t)) \cap B^n = \text{def}(h^*(t)) \cap B^0 = \text{def}(h^*(t)) \cap 1 \stackrel{(6)}{=} 1 \stackrel{(7)}{\subseteq} \text{def}(h_{n+1}^*(t)).$$

Case 2.2:  $n > 0$ . Let  $w \in \text{def}(h^*(t)) \cap B^n$ . By (6),  $w \in \text{def}(c(h^*(u)))$  and thus  $w = bv$  for some  $b \in B$  and  $v \in \text{def}(h^*(u)) \cap B^{n-1}$ . By induction hypothesis,  $v \in \text{def}(h_n^*(u))$ . Hence

$$w = bv \in \text{def}(c(h_n^*(u))) \stackrel{(7)}{=} \text{def}(h_{n+1}^*(t)).$$

Therefore, (4) holds true in both subcases.

Case 3:  $t \in I$ . Then  $\text{def}(h^*(t)) \cap B^n = \text{def}(t) \cap B^n = 1 = \text{def}(t) = \text{def}(h_{n+1}^*(t))$ .

Case 4:  $t = c(u)$  for some  $c : e \rightarrow s \in C$  and  $u \in T_\Sigma(I + V)_e$ . Then

$$h^*(t) = h^*(c(u)) = c^A(h^*(u)) = c(h^*(u)), \quad (8)$$

$$h_{n+1}^*(t) = h_{n+1}^*(c(u)) = c^A(h_{n+1}^*(u)) = c(h_{n+1}^*(u)) \quad (9)$$

Case 4.1:  $n = 0$ . Then

$$\text{def}(h^*(t)) \cap B^n = \text{def}(h^*(t)) \cap B^0 = \text{def}(h^*(t)) \cap 1 \stackrel{(8)}{=} 1 \stackrel{(9)}{\subseteq} \text{def}(h_{n+1}^*(t)).$$

Case 4.2:  $n > 0$ . Let  $w \in \text{def}(h^*(t)) \cap B^n$ . By (8),  $w \in \text{def}(c(h^*(u)))$  and thus  $w = bv$  for some  $b \in B$  and  $v \in \text{def}(h^*(u)) \cap B^{n-1}$ . By induction hypothesis,  $v \in \text{def}(h_n^*(u))$ . Since  $h_n \leq h_{n+1}$  and  $CT_\Sigma^\perp(I) \in \text{Poset}^S$ ,  $h_n^* \leq h_{n+1}^*$ . Hence

$$w = bv \in \text{def}(c(h_n^*(u))) \subseteq \text{def}(c(h_{n+1}^*(u))) \stackrel{(9)}{=} \text{def}(h_{n+1}^*(t)).$$

Therefore, (4) holds true in both subcases.

*Case 5:*  $t = i(u) \in T_\Sigma(I + V)_e$  for some  $e = \coprod_{i \in I} e_i$ ,  $i \in I$  and  $u \in T_\Sigma(I + V)_{e_i}$ . Then we obtain (4) as in Case 4 with  $i$  instead of  $c$ .

*Case 6:*  $t = ()\{i \rightarrow t_i \mid i \in I\} \in T_\Sigma(I + V)_e$  for some  $e = \prod_{i \in I} e_i$  and  $(t_i)_{i \in I} \in \bigtimes_{i \in I} T_\Sigma(I + V)_{e_i}$ . Then for all  $i \in I$ ,

$$\pi_i(h^*(t)) = h^*(t_i) = \pi_i(()\{i \rightarrow h^*(t_i) \mid i \in I\}),$$

$$\pi_i(h_{n+1}^*(t)) = \pi_i(()\{i \rightarrow h_{n+1}^*(t_i) \mid i \in I\}).$$

Hence

$$h^*(t) = ()\{i \rightarrow h^*(t_i) \mid i \in I\}, \tag{10}$$

$$h_{n+1}^*(t) = ()\{i \rightarrow h_{n+1}^*(t_i) \mid i \in I\} \tag{11}$$

*Case 6.1:*  $n = 0$ . Then

$$\text{def}(h^*(t)) \cap B^n = \text{def}(h^*(t)) \cap B^0 = \text{def}(h^*(t)) \cap 1 \stackrel{(10)}{=} 1 \stackrel{(11)}{\subseteq} \text{def}(h_{n+1}^*(t)).$$

*Case 6.2:*  $n > 0$ . Let  $w \in \text{def}(h^*(t)) \cap B^n$ . By (10),  $w \in \text{def}(()\{i \rightarrow h^*(t_i) \mid i \in I\})$  and thus  $w = iv$  for some  $i \in I$  and  $v \in \text{def}(h^*(t_i)) \cap B^{n-1}$ . By induction hypothesis,  $v \in \text{def}(h_{n+1}^*(t_i))$ . Since  $h_n \leq h_{n+1}$  and  $\text{CT}_\Sigma^\perp(I) \in \text{Poset}^S$ ,  $h_n^* \leq h_{n+1}^*$ . Hence

$$w = iv \in \text{def}(()\{i \rightarrow h_n^*(t_i) \mid i \in I\}) \subseteq \text{def}(()\{i \rightarrow h_{n+1}^*(t_i) \mid i \in I\}) \stackrel{(11)}{=} \text{def}(h_{n+1}^*(t)).$$

Therefore, (4) holds true in both subcases. □

## The coalgebraic approach for computing solutions of term equations

Let  $E : V \rightarrow T_\Sigma(V)$  be a system of iterative  $\Sigma$ -equations without input and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ . Then the above step function  $E^{\mathcal{A}}$  reduces to:

$$\begin{aligned} E^{\mathcal{A}} : A^V &\rightarrow A^V \\ g &\mapsto g^* \circ E, \end{aligned}$$

and  $g \in A^V$  solves  $E$  in  $\mathcal{A}$  iff  $g^* \circ E = g$ .

**Lemma ITER** Let  $E, E' : V \rightarrow T_\Sigma(V)$  be systems of iterative  $\Sigma$ -equations and  $\mathcal{A}, \mathcal{B}$  be  $\Sigma$ -algebras with carriers  $A$  and  $B$ , respectively.

(i) For all  $\Sigma$ -homomorphisms  $f : \mathcal{A} \rightarrow \mathcal{B}$  and  $g \in A^V$ ,

$$f \circ E^{\mathcal{A}}(g) = E^{\mathcal{B}}(f \circ g).$$

(ii) For all strict and  $\omega$ -continuous  $\Sigma$ -homomorphisms  $f : \mathcal{A} \rightarrow \mathcal{B}$ ,

$$f \circ (E^{\mathcal{A}})^\infty = (E^{\mathcal{B}})^\infty.$$

(iii) Suppose that for all  $x \in V$ ,  $\mathcal{A}$  satisfies the equation  $E(x) = E'(x)$ . Then  $E$  and  $E'$  have the same solutions.

*Proof of (i).*

$$f \circ E^{\mathcal{A}}(g) \stackrel{\text{Def. } E^{\mathcal{A}}}{=} f \circ g^* \circ E \stackrel{\text{Lemma } \textcolor{violet}{SUBST}}{=} (f \circ g)^* \circ E \stackrel{\text{Def. } E^{\mathcal{B}}}{=} E^{\mathcal{B}}(f \circ g).$$

*Proof of (ii).* First we show

$$f \circ (E^{\mathcal{A}})^n(\lambda x. \perp_A) = (E^{\mathcal{B}})^n(\lambda x. \perp_B) \quad (4)$$

for all  $n \in \mathbb{N}$  by induction on  $n$ . Since  $f$  is strict,

$$f \circ (E^{\mathcal{A}})^0(\lambda x. \perp^A) = f \circ (\lambda x. \perp_A) = \lambda x. \perp_B = (E^{\mathcal{B}})^0(\lambda x. \perp_B).$$

If  $n > 0$ , then by (i),

$$\begin{aligned} f \circ (E^{\mathcal{A}})^n(\lambda x. \perp^A) &= f \circ E^{\mathcal{A}}((E^{\mathcal{A}})^{n-1}(\lambda x. \perp^A)) \stackrel{(i)}{=} E^{\mathcal{B}}(f \circ (E^{\mathcal{A}})^{n-1}(\lambda x. \perp^A)) \\ &\stackrel{\text{ind. hyp.}}{=} E^{\mathcal{B}}((E^{\mathcal{B}})^{n-1}(\lambda x. \perp_B)) = (E^{\mathcal{B}})^n(\lambda x. \perp_B). \end{aligned}$$

Hence (4) holds true, and we conclude (ii) as follows:

$$\begin{aligned} f \circ (E^{\mathcal{A}})^\infty &= f \circ \bigsqcup_{n \in \mathbb{N}} (E^{\mathcal{A}})^n(\lambda x. \perp_A) \stackrel{f \text{ } \omega\text{-continuous}}{=} \bigsqcup_{n \in \mathbb{N}} (f \circ (E^{\mathcal{A}})^n(\lambda x. \perp_A)) \\ &\stackrel{(4)}{=} \bigsqcup_{n \in \mathbb{N}} (E^{\mathcal{B}})^n(\lambda x. \perp_B) = (E^{\mathcal{B}})^\infty. \end{aligned}$$

*Proof of (iii).* W.l.o.g. let  $g \in A^V$  solve  $E$  in  $\mathcal{A}$ . Then  $g^*(E'(x)) = g^*(E(x)) = g(x)$ , i.e.,  $g$  solves  $E'$  as well.  $\square$

Theorem SOLC can also be derived from the finality of  $CT_\Sigma$  in  $Alg_{co\Sigma}$  (see [From constructors to destructors](#) in chapter 12).

For this purpose,  $T_\Sigma(V)$  is turned into the  $co\Sigma$ -algebra  $\textcolor{red}{T}_{\Sigma,E}$  is defined as follows:

- For all  $s \in S$ ,  $\textcolor{blue}{T}_{\Sigma,E}(s) = T_\Sigma(V)_s$ .
- For all  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(V)_e$ ,  $\textcolor{blue}{d}_s^{T_{\Sigma,E}}(c(t)) = c(t) \in \coprod_{c:e \rightarrow s} T_\Sigma(V)_e$ .
- For all  $s \in S$  and  $x \in V_s$ ,  $E(x) = c(t)$  implies  $\textcolor{blue}{d}_s^{T_{\Sigma,E}}(x) = c(t)$ .

## Theorem SOLD

$E^\dagger =_{def} V \xrightarrow{inc_V} T_\Sigma(V) \xrightarrow{unfold^{T_{\Sigma,E}}} CT_\Sigma$  solves  $E$  in  $CT_\Sigma$  uniquely. Moreover,

$$unfold^{T_{\Sigma,E}} \circ inc_{T_\Sigma} = fold^{CT_\Sigma} : T_\Sigma \rightarrow CT_\Sigma. \quad (1)$$

Since  $CT_\Sigma$  is final in  $Alg_{co\Sigma}$ ,  $E^\dagger$  yields a unique solution in every final  $co\Sigma$ -algebra.

*Proof.* First we show that the  $co\Sigma$ -homomorphism  $unfold^{T_{\Sigma,E}} : T_{\Sigma}(V) \rightarrow CT_{\Sigma}$  is also  $\Sigma$ -homomorphic. Let  $c : e \rightarrow s \in C$  and  $t \in T_{\Sigma}(V)_e$ .

Since  $d_s^{T_{\Sigma,E}}(c(t)) = c(t) = \iota_c(t)$ , the definition of  $unfold^{T_{\Sigma,E}}$  (see From constructors to destructors) implies

$$unfold_s^{T_{\Sigma,E}}(c(t)) = c(unfold_e^{T_{\Sigma,E}}(t)). \quad (2)$$

Hence

$$unfold_s^{T_{\Sigma,E}}(c^{T_{\Sigma,E}}(t)) = unfold_s^{T_{\Sigma,E}}(c(t)) \stackrel{(2)}{=} c(unfold_e^{T_{\Sigma,E}}(t)) = c^{CT_{\Sigma}}(unfold_e^{T_{\Sigma,E}}(t)).$$

Therefore,  $unfold^{T_{\Sigma,E}}$  is  $\Sigma$ -homomorphic and thus by the definition of  $E^{\dagger}$ ,

$$unfold^{T_{\Sigma,E}} = (E^{\dagger})^* \quad (3)$$

because there is only one  $\Sigma$ -homomorphism  $h : T_{\Sigma,E} \rightarrow CT_{\Sigma}$  with  $h \circ inc_V = E^{\dagger}$ .

Let  $x \in V$ ,  $c : e \rightarrow s \in C$ ,  $t \in T_{\Sigma}(V)_e$  and  $E(x) = c(t)$ . Then  $d_s^{T_{\Sigma,E}}(x) = d_s^{T_{\Sigma,E}}(c(t)) = c(t) = \iota_c(t)$  and thus, again by the definition of  $unfold^{T_{\Sigma,E}}$ ,

$$unfold_s^{T_{\Sigma,E}}(x) = c(unfold_e^{T_{\Sigma,E}}(t)). \quad (4)$$

Hence

$$(E^\dagger)_s^*(E(x)) = (E^\dagger)_s^*(c(t)) \stackrel{(3)}{=} \text{unfold}_s^{T_{\Sigma,E}}(c(t)) \stackrel{(2)}{=} c(\text{unfold}_e^{T_{\Sigma,E}}(t)) \stackrel{(4)}{=} \text{unfold}_s^{T_{\Sigma,E}}(x) \\ = E^\dagger(x),$$

i.e.,  $E^\dagger$  solves  $E$  in  $CT_\Sigma$ .

Let  $g : V \rightarrow CT_\Sigma$  solve  $E$  in  $CT_\Sigma$ .

First we show that the  $\Sigma$ -homomorphism  $g^* : T_{\Sigma,E} \rightarrow CT_\Sigma$  is also  $co\Sigma$ -homomorphic.

Let  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(V)_e$ . Then

$$d_s^{CT_\Sigma}(g_s^*(c(t))) = d_s^{CT_\Sigma}(c(g_e^*(t))) = c(g_e^*(t)) = g_s^*(c(t)) = g_s^*(d_s^{T_{\Sigma,E}}(c(t))). \quad (5)$$

Let  $x \in V$ ,  $c : e \rightarrow s \in C$ ,  $t \in T_\Sigma(V)_e$  and  $E(x) = c(t)$ . Then

$$d_s^{CT_\Sigma}(g_s^*(x)) = d_s^{CT_\Sigma}(g_s(x)) \stackrel{g \text{ solves } E}{=} d_s^{CT_\Sigma}(g^*(E(x))) = d_s^{CT_\Sigma}(g_s^*(c(t))) \\ \stackrel{(5)}{=} g_e^*(d_s^{T_{\Sigma,E}}(c(t))) = g_e^*(d_s^{T_{\Sigma,E}}(E(x))) = g_e^*(d_s^{T_{\Sigma,E}}(x)). \quad (6)$$

By (5) and (6),  $g^*$  is  $co\Sigma$ -homomorphic.

Suppose that  $g, h : V \rightarrow CT_\Sigma$  solve  $E$  in  $CT_\Sigma$ . Since  $g^*$  and  $h^*$  are  $co\Sigma$ -homomorphic and thus agree with each other because  $CT_\Sigma$  is final in  $Alg_{co\Sigma}$ . Hence

$$g = g^* \circ inc_V = h^* \circ inc_V = h.$$

*Proof of (1):* The restriction  $\mathcal{B}$  of  $T_{\Sigma,E}$  to  $T_\Sigma$  is a  $co\Sigma$ -subalgebra of  $T_{\Sigma,E}$ . Hence the inclusion  $inc_{T_\Sigma} : T_\Sigma \rightarrow T_\Sigma(V)$  is  $co\Sigma$ -homomorphic and thus  $unfold^{T_{\Sigma,E}} \circ inc_{T_\Sigma} = unfold^{\mathcal{B}}$  because  $CT_\Sigma$  is final in  $Alg_{co\Sigma}$ . Above we have shown that  $unfold^{T_{\Sigma,E}}$  is  $\Sigma$ -homomorphic. Hence  $unfold^{\mathcal{B}} = unfold^{T_{\Sigma,E}} \circ inc_{T_\Sigma}$  is also  $\Sigma$ -homomorphic. Therefore,  $unfold^{\mathcal{B}} = fold^{CT_\Sigma}$  because  $T_\Sigma$  is initial in  $Alg_\Sigma$ .  $\square$

A  $\Sigma$ -term that is representable as a component of the unique solution in  $CT_\Sigma$  of a finite system of iterative  $\Sigma$ -equations is rational (see chapter 2).

By Theorem SOLC (with  $I = \emptyset$ ), the equation  $g^* \circ E = g$  has exactly one solution  $g : V \rightarrow CT_\Sigma$ , namely  $E_C^\infty$ . Hence  $E^\dagger = E_C^\infty$  and thus triangle (7) in the following diagram commutes.

Let  $\mathcal{A}$  be an  $\omega$ -continuous  $\Sigma$ -algebra with carrier  $A$  and  $fold_\omega^{\mathcal{A}}$  be defined as in the proof of Theorem CFREE.

Since  $fold_\omega^{\mathcal{A}}$  is  $\Sigma$ -homomorphic and  $E_C^\infty$  agrees with  $(E^{CT_\Sigma^\perp})^\infty$ , (8) follows from Lemma ITER (ii):

$$\begin{array}{ccc}
 V & \xrightarrow{(E^{\mathcal{A}})^{\infty}} & \mathcal{A} \\
 inc_V \downarrow & \searrow (7) & \uparrow fold_{\omega}^{\mathcal{A}} \\
 T_{\Sigma}(V) & \xrightarrow{(8)} & CT_{\Sigma}^{\perp} \\
 & \xrightarrow{unfold^{T_{\Sigma}, E}} &
 \end{array}$$

## Corollary RECITER

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive signature,  $\Sigma' = (S, \mathcal{I}, C)$  be a constructive polynomial signature,  $E : V \rightarrow T_{\Sigma'}(V)$  be a system of iterative  $\Sigma'$ -equations,

$$C_V = \{x : 1 \rightarrow s \mid x \in V_s, s \in S\},$$

$\Sigma'' = (S, \mathcal{I}, C \cup C_V)$  and for all  $c : e \rightarrow s \in C \cup C_V$  and  $d : s \rightarrow e' \in D$ , let  $op_{c,d} : e \rightarrow e' \in admitted(D)$  (see Recursive functions).

Theorem RECFUN3 provides an extension of every final  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  to a  $\Sigma'$ -algebra.

For all  $g \in A^V$ , let  $\mathcal{A}_g$  be the  $\Sigma''$ -algebra with  $\mathcal{A}_g|_{\Sigma'} = \mathcal{A}$  and  $x^{\mathcal{A}_g} = g(x)$  for all  $x \in V$ .

If for all solutions  $g \in A^V$  of  $E$ ,  $\mathcal{A}_g$  satisfies

$$\text{rec}(E) = \{c = \text{obj}\{d.\text{op}_{c,d}\}_{d:s \rightarrow e' \in D} \mid c : 1 \rightarrow s \in C_V\},$$

then  $E$  has at most one solution in  $\mathcal{A}$ .

*Proof.* Let  $g, h \in A^V$  solve  $E$  in  $\mathcal{A}$ . Since  $\mathcal{A}$  is a final  $\Sigma$ -algebra, Theorem RECFUN3 implies that there is a *unique* extension of  $\mathcal{A}$  to a  $\Sigma''$ -algebra that satisfies

$$E' = \{c = \text{obj}\{d.\text{op}_{c,d}\}_{d:s \rightarrow e' \in D} \mid c : e \rightarrow s \in C \cup C_V\}.$$

By assumption, both  $\mathcal{A}_g$  and  $\mathcal{A}_h$  satisfy  $\text{rec}(E)$ . Moreover,  $\mathcal{A}$  and thus  $\mathcal{A}_g$  and  $\mathcal{A}_h$  satisfy  $E' \setminus \text{rec}(E)$ . Hence both  $\mathcal{A}_g$  and  $\mathcal{A}_h$  satisfy  $E'$  and thus agree with each other. In particular, for all  $x \in V$ ,  $g(x) = x^{\mathcal{A}_g} = x^{\mathcal{A}_h} = h(x)$ .  $\square$

We return to the general case where  $I$  may be nonempty,  $E$  maps  $V$  to  $T_\Sigma(I + V)$  and  $E^\mathcal{A}$  is an endofunction on  $A^I \rightarrow A^V$ .

Let  $\Sigma(I) = (S, C \cup \{\text{val}_s : I_s \rightarrow s \mid s \in S\})$  (see [Term folding](#)),  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ ,  $g \in A^I$  and  $\mathcal{A}^g$  be the  $\Sigma(I)$ -algebra with  $\mathcal{A}^g|_\Sigma = \mathcal{A}$  and  $\text{val}_s^{\mathcal{A}^g} = g_s$  for all  $s \in S$ .

For all  $g \in CT_{\Sigma}^I$ , the  $\Sigma(I)$ -homomorphism  $\textcolor{red}{g}' : CT_{\Sigma(I)} \rightarrow CT_{\Sigma}^g$  is defined as follows:

- For all  $s \in S$  and  $i \in I_s$ ,  $\textcolor{blue}{g}'(\text{val}_s(i)) = g_s(i)$ .
- For all  $t = x\{i \rightarrow t_i \mid i \in I\} \in CT_{\Sigma(I)}$  with  $x \notin \{\text{val}_s \mid s \in S\}$ ,

$$\textcolor{blue}{g}'(t) = x\{i \rightarrow \textcolor{blue}{g}'(t_i) \mid i \in I\}.$$

The  $S$ -sorted substitution  $\textcolor{red}{\sigma} : I + V \rightarrow T_{\Sigma(I)}(V)$  assigns  $x$  to all  $x \in V$  and  $\text{val}_s(i)$  to all  $i \in I_s$ ,  $s \in S$ .

## Theorem SOLI

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .

- (1) If  $f : A^I \rightarrow A^V$  solves  $E$  in  $\mathcal{A}$ , then for all  $g \in A^I$ ,  $f(g)$  solves  $\sigma^* \circ E$  in  $\mathcal{A}^g$ .
- (2) Let  $\mathcal{B}$  be a  $\Sigma(I)$ -algebra whose carrier includes  $A$ . If  $h \in A^V$  solves  $\sigma^* \circ E$  in  $\mathcal{B}$ , then

$$[\text{val}^{\mathcal{B}}, h]^* \circ E = h.$$

- (3)  $f : A^I \rightarrow A^V$  solves  $E$  in  $\mathcal{A}$  iff for all  $g \in A^I$ ,  $f(g)$  solves  $\sigma^* \circ E$  in  $\mathcal{A}^g$ . In particular, for all  $g \in A^I$ ,

$$(E^{\mathcal{A}})^{\infty}(g) = ((\sigma^* \circ E)^{\mathcal{A}^g})^{\infty}.$$

$$(4) \quad \begin{aligned} E^\dagger : CT_\Sigma^I &\rightarrow CT_\Sigma^V \\ g &\mapsto V \xrightarrow{\text{inc}_V} T_{\Sigma(I)}(V) \xrightarrow{\text{unfold}^{T_{\Sigma(I)}, \sigma^* \circ E}} CT_{\Sigma(I)} \xrightarrow{g'} CT_\Sigma^g \end{aligned}$$

solves  $E$  in  $CT_\Sigma$  uniquely.

Since  $CT_\Sigma$  is final in  $\text{Alg}_{co\Sigma}$ , (4) implies that  $E$  has a unique solution  $E^\dagger$  in every final  $co\Sigma$ -algebra  $\mathcal{A}$  and thus  $(\pi_x \circ E^\dagger)_{x \in V}$  is the unique tuple  $(f_x : A^I \rightarrow A)_{x \in V}$  of functions that satisfies

$$f_x(g) = E(x)[(f_x(g)/x \mid x \in V)][g(i) \mid i \in I]$$

for all  $x \in V$  and  $g \in A^I$  where  $A$  is the carrier of  $\mathcal{A}$ .

*Proof.* Let  $\mathcal{B}$  be a  $\Sigma(I)$ -algebra whose carrier includes  $A$  and  $h \in A^V$ . First we show

$$h^* \circ \sigma^* = [val^{\mathcal{B}}, h]^* : T_\Sigma(I + V) \rightarrow A \quad (4)$$

by induction on  $T_\Sigma(I + V)$ : For all  $x \in V$ ,

$$h^*(\sigma^*(x)) = h^*(\sigma(x)) = h^*(x) = h(x) = [val^{\mathcal{B}}, h](x) = [val^{\mathcal{B}}, h]^*(x).$$

For all  $i \in I$ ,

$$h^*(\sigma^*(i)) = h^*(\sigma(i)) = h^*(val(i)) = val^{\mathcal{B}}(h^*(i)) = val^{\mathcal{B}}(i) = [val^{\mathcal{B}}, h](i) = [val^{\mathcal{B}}, h]^*(i).$$

For all  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(I + V)_e$ ,

$$h^*(\sigma^*(c(t))) = h^*(c(\sigma^*(t))) = c^{\mathcal{B}}(h^*(\sigma^*(t))) \stackrel{\text{ind. hyp.}}{=} c^{\mathcal{B}}([val^{\mathcal{B}}, h]^*(t)) = [val^{\mathcal{B}}, h]^*(c(t)).$$

For all sum types  $e = \coprod_{i \in J} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in J$  and  $t \in T_\Sigma(I + V)_{e_i}$ ,

$$h^*(\sigma^*(i(t))) = h^*(i(\sigma^*(t))) = \iota_i(h^*(\sigma^*(t))) \stackrel{\text{ind. hyp.}}{=} \iota_i([val^\mathcal{B}, h]^*(t)) = [val^\mathcal{B}, h]^*(c(i(t))).$$

For all product types  $e = \prod_{i \in J} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t = ()\{i \rightarrow t_i \mid i \in J\} \in T_\Sigma(I + V)_e$ ,

$$\begin{aligned} \pi_i(h^*(\sigma^*(t))) &= \pi_i(h^*(\{i \rightarrow \sigma^*(t_i) \mid i \in J\})) = h^*(\sigma^*(t_i)) \stackrel{\text{ind. hyp.}}{=} [val^\mathcal{B}, h]^*(t_i) \\ &= \pi_i(()\{i \rightarrow [val^\mathcal{B}, h]^*(t_i) \mid i \in J\}) = [val^\mathcal{B}, h]^*(t). \end{aligned}$$

*Proof of (1).* Suppose that  $f$  solves  $E$  in  $\mathcal{A}$ . Then for all  $g \in A^I$ ,

$$f(g)^* \circ \sigma^* \circ E \stackrel{(4)}{=} [val^{\mathcal{A}^g}, f(g)]^* \circ E = [g, f(g)]^* \circ E = f(g),$$

i.e.,  $f(g)$  solves  $\sigma^* \circ E$  in  $\mathcal{A}^g$ .

*Proof of (2).* Suppose that  $h \in A^V$  solves  $\sigma^* \circ E$  in  $\mathcal{B}$ . Then

$$[val^\mathcal{B}, h]^* \circ E \stackrel{(4)}{=} h^* \circ \sigma^* \circ E = h.$$

(1) and (2) imply (3).

*Proof of (4).* By Theorem SOLC or SOLD,

$$h = (\sigma^* \circ E)_C^\infty \quad \text{and} \quad h = \text{unfold}^{T_{\Sigma(I)}, \sigma^* \circ E} \circ \text{inc}_V$$

solve  $\sigma^* \circ E$  in  $CT_{\Sigma(I)}$ . Hence by (2), for all  $g \in CT_\Sigma^I$ ,

$$[g, h]^* \circ E = [\text{val}^{CT_{\Sigma(I)}^g}, h]^* \circ E = h. \quad (5)$$

Since  $g'$  is  $\Sigma(I)$ -homomorphic,

$$\begin{aligned} g' \circ h &\stackrel{(5)}{=} g' \circ [g, h]^* \circ E \stackrel{\text{Lemma } \textcolor{violet}{SUBST}}{=} (g' \circ [g, h])^* \circ E = [g' \circ g, g' \circ h]^* \circ E \\ &\stackrel{\text{img}(g) \subseteq CT_\Sigma}{=} [g, g' \circ h]^* \circ E, \end{aligned}$$

i.e.,  $f : CT_\Sigma^I \rightarrow CT_\Sigma^V$  with  $f(g) = g' \circ h$  solves  $E$  in  $CT_\Sigma$ .

Suppose that  $f, f' : CT_\Sigma^I \rightarrow CT_\Sigma^V$  solve  $E$  in  $CT_\Sigma$ . By (1), for all  $g \in CT_\Sigma^I$ ,  $f(g)$  and  $f'(g)$  solve  $\sigma^* \circ E$  in  $CT_\Sigma^g$ . Hence by Theorem SOLC (or SOLD),  $f(g) = f'(g)$ . Therefore,  $E$  has at most one solution in  $CT_\Sigma$ .  $\square$

## Examples

1. Let  $\Sigma = coStream(X)$ ,  $I = \{x, y\}$ ,  $V = \{blink\}$  and

$$\begin{aligned} E : V &\rightarrow T_\Sigma(I + V) \\ \text{blink} &\mapsto \text{cons}(x, \text{cons}(y, \text{blink})). \end{aligned}$$

The unique solution  $g : CT_\Sigma^I \rightarrow CT_\Sigma^V$  of  $E$  in  $CT_\Sigma$  is defined as follows:

For all  $h \in CT_\Sigma^I$  and  $w \in \{\epsilon, 1, 2\}^*$ ,

$$g(h)(\text{blink})(w) = \begin{cases} \text{cons} & \text{if } w \in (\epsilon 2)^*, \\ h(x) & \text{if } \exists n \in \mathbb{N} : w = (\epsilon 2)^n \epsilon 1 \wedge \text{even}(n), \\ h(y) & \text{if } \exists n \in \mathbb{N} : w = (\epsilon 2)^n \epsilon 1 \wedge \text{odd}(n), \\ \perp & \text{otherwise.} \end{cases}$$

2. Let  $\Sigma = coDAut(\mathbb{Z}, 2)$ ,  $V = \{esum, osum\}$  and

$$E : V \rightarrow T_\Sigma(V)$$

$$\text{esum} \mapsto \text{new}(()\{\delta \rightarrow ()\{x \triangleright \text{even}(x) \rightarrow \text{esum}, x \triangleright \text{odd}(x) \rightarrow \text{osum} \mid x \in \mathbb{Z}\}, \beta \rightarrow 1\})$$

$$\text{osum} \mapsto \text{new}(()\{\delta \rightarrow ()\{x \triangleright \text{even}(x) \rightarrow \text{osum}, x \triangleright \text{odd}(x) \rightarrow \text{esum} \mid x \in \mathbb{Z}\}, \beta \rightarrow 0\}).$$

The unique solution  $g : V \rightarrow CT_{\Sigma}$  of  $E$  in  $CT_{\Sigma}$  is defined as follows:

For all  $w \in (\{\epsilon, \delta, \beta\} \cup \mathbb{Z})^*$ ,

$$g(esum)(w) = \begin{cases} new & \text{if } \exists n \in \mathbb{N}, x_1, \dots, x_n \in \mathbb{Z} : w \in (\epsilon\delta\mathbb{Z})^*, \\ 1 & \text{if } \exists n \in \mathbb{N}, x_1, \dots, x_n \in \mathbb{Z} : \\ & w = \epsilon\delta x_1 \dots \epsilon\delta x_n \epsilon\beta \wedge even(x_1 + \dots + x_n), \\ 0 & \text{if } \exists n \in \mathbb{N}, x_1, \dots, x_n \in \mathbb{Z} : \\ & w = \epsilon\delta x_1 \dots \epsilon\delta x_n \epsilon\beta \wedge odd(x_1 + \dots + x_n), \\ \perp & \text{otherwise.} \end{cases}$$

Here  $\delta$  and  $\beta$  serve as indices of the domain  $state^X \times 2$  of  $new$  because 1 and 2 would conflict with integer numbers that also occur as edge labels here.

$CT_{\Sigma}$  is a  $DAut(\mathbb{Z}, 2)$ -algebra and  $\{g(esum), g(osum)\}$  is the carrier of a  $DAut(\mathbb{Z}, 2)$ -subalgebra of  $CT_{\Sigma}$  that is isomorphic to the sample  $DAut(\mathbb{Z}, 2)$ -algebra 7 in chapter 9.

3. Let  $S = \{cmd, exp, bexp\}$ ,  $X$  be a set of program variables that take values in some set  $Val$  of values,

$$\begin{aligned} F = \{ & \ skip : 1 \rightarrow cmd, \ assign : X \times exp \rightarrow cmd, \\ & \ seq : cmd \times cmd \rightarrow cmd, \\ & \ cond : bexp \times cmd \times cmd \rightarrow cmd \}, \end{aligned}$$

$\Sigma = (S, \mathcal{I}, F)$ ,  $I = \{b, c\}$ ,  $V = \{loop\}$  and

$$\begin{aligned} E : V & \rightarrow T_\Sigma(I + V) \\ loop & \mapsto cond(b, seq(c, loop), skip) \end{aligned}$$

The unique solution  $g : CT_\Sigma^I \rightarrow CT_\Sigma^V$  of  $E$  in  $CT_\Sigma$  is defined as follows:

For all  $h \in CT_\Sigma^I$  and  $w \in \{\epsilon, 1, 2, 3\}^*$ ,

$$g(h)(loop)(w) = \begin{cases} cond & \text{if } w \in (\epsilon 2 \epsilon 2)^*, \\ h(b)(w'') & \text{if } \exists w', w'' : w = w'w'' \wedge w' \in (\epsilon 2 \epsilon 2)^* \epsilon 1, \\ seq & \text{if } w \in (\epsilon 2 \epsilon 2)^* \epsilon 2, \\ skip & \text{if } w \in (\epsilon 2 \epsilon 2)^* \epsilon 3, \\ h(c)(w'') & \text{if } \exists w', w'' : w = w'w'' \wedge ((\epsilon 2 \epsilon 2)^*) \epsilon 2 \epsilon 1, \\ \perp & \text{otherwise.} \end{cases}$$

Let  $\mathcal{A}$  be the  $\omega$ -continuous  $\Sigma$ -algebra of “store states” that is defined as follows:

Let  $St = Val^X$ . For all  $f : St \rightarrow 2$ ,  $g, h : St \rightarrow St + 1$ ,  $e : St \rightarrow Val$ ,  $x \in X$  and  $st \in St$ ,

$$\begin{aligned}\mathcal{A}(cmd) &= St \rightarrow St + 1, \\ \mathcal{A}(exp) &= St \rightarrow Val, \\ \mathcal{A}(bexp) &= St \rightarrow 2, \\ skip^{\mathcal{A}} &= id_{St}, \\ assign^{\mathcal{A}}(x, e)(st) &= st[e(st)/x], \\ seq^{\mathcal{A}}(g, h) &= h \circ g, \\ cond^{\mathcal{A}}(f, g, h)(st) &= \text{if } f(st) = 1 \text{ then } g(st) \text{ else } h(st).\end{aligned}$$

The least solution of  $E$  in  $\mathcal{A}$  provides the usual semantics of the while-loop operator  $while : bexp \times cmd \rightarrow cmd$ : For all  $f : St \rightarrow 2$  and  $g : St \rightarrow St + 1$ ,

$$while^{\mathcal{A}}(f, g) = (E'^{\mathcal{A}^h})^\infty$$

where  $h \in A^I$  maps  $b$  to  $f$  and  $c$  to  $g$ .

□

## Flowchart equations

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature and  $V, O$  be  $S$ -sorted sets of “internal” and “output variables”, respectively. An  $S$ -sorted function

$$E : V \rightarrow \overline{T_\Sigma}(V + O)$$

is called a **system of iterative  $\Sigma$ -equations** if  $\text{img}(E) \cap (V + O) = \emptyset$  (see  $\Sigma$ -flowcharts).

Hence for all  $s \in S$  and  $x \in V_s$ ,  $E(x) = d(t)$  for some  $d : s \rightarrow e \in D$  and  $t \in \overline{T_\Sigma}(V + O)_e$ .

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .

$f : V \times A \rightarrow O \times A$  solves  $E$  in  $\mathcal{A}$  if

$$[\text{curry}(f), \text{return}_O]^+ \circ E = \text{curry}(f).$$

Hence  $f$  solves  $E$  in  $\mathcal{A}$  iff  $f$  is the fixpoint of the following step function:

$$\begin{aligned} E^{\mathcal{A}} : (V \times A \rightarrow O \times A) &\rightarrow (V \times A \rightarrow O \times A) \\ f &\mapsto \text{uncurry}([\text{curry}(f), \text{return}_O]^+ \circ E) \end{aligned}$$

Hence solutions of flowchart equations are somewhat dual to solutions  $f : A^I \rightarrow A^V$  of term equations (see above).

Let  $\mathcal{A}$  be  $\omega$ -continuous. According to the section [Continuous algebras](#), the partial orders, least elements and suprema of  $A$  can be lifted to  $O \times A$  and then to  $V \times A \rightarrow O \times A$ , i.e.,  $O \times A$  and  $V \times A \rightarrow O \times A$  are  $\omega$ -CPOs.

$E^{\mathcal{A}}$  is  $\omega$ -continuous. Hence by [Kleene's Fixpoint Theorem](#) (1),

$$(E^{\mathcal{A}})^\infty = \bigsqcup_{n<\omega} (E^{\mathcal{A}})^n(\perp) : V \times A \rightarrow O \times A$$

is the least fixpoint of  $E^{\mathcal{A}}$  where  $\perp : V \times A \rightarrow O \times A$  maps all elements of  $V \times A$  to the least element of  $O \times A$ , which maps all elements of  $A$  to the least element of the sum (!)  $O \times A$ .

**Lemma FEQ** For all  $g : V \rightarrow \overline{CT_{\Sigma}}(O)$ ,

$$[g, inc_O]^* \circ E = g \Rightarrow f_g \text{ solves } E \text{ in } \mathcal{A}$$

where  $f_g : V \times A \rightarrow O \times A$  maps  $(x, a) \in V \times A$  to  $(return_O^+)_\omega(g(x))(a) \in O \times A$  (see [Continuous algebras](#)).

*Proof.* By definition,

$$curry(f_g) = (return_O^+)_\omega \circ g. \quad (1)$$

Suppose that

$$[g, inc_O]^* \circ E = g. \quad (2)$$

Then

$$\begin{aligned} [curry(f_g), return_O]^+ \circ E &\stackrel{(1)}{=} [(return_O^+)_\omega \circ g, return_O]^+ \circ E \\ &= [(return_O^+)_\omega \circ g, return_O^+ \circ inc_O]^+ \circ E \\ &= [(return_O^+)_\omega \circ g, (return_O^+)_\omega \circ inc_O]^+ \circ E = ((return_O^+)_\omega \circ [g, inc_O])^+ \circ E \\ &\stackrel{\text{Lemma } SUBSTFC}{=} (return_O^+)_\omega \circ [g, inc_O]^* \circ E \stackrel{(2)}{=} (return_O^+)_\omega \circ g \stackrel{(1)}{=} curry(f_g), \end{aligned}$$

i.e.,  $f_g$  solves  $E$  in  $\mathcal{A}$ . □

### Theorem SOLF (“dualization” of Theorem SOLC)

Let  $E : V \rightarrow \overline{T_\Sigma}(V + O)$  be a system of iterative  $\Sigma$ -equations. There is exactly one  $g : V \rightarrow \overline{CT_\Sigma}(O)$  with (2).

*Proof.* Define the step function  $E_C : \overline{CT_\Sigma^\perp}(O)^V \rightarrow \overline{CT_\Sigma^\perp}(O)^V$  follows:

For all  $g : V \rightarrow \overline{CT_\Sigma^\perp}(O)$ ,  $E_C(g) = [g, inc_O]^* \circ E$ . Since  $E_C$  is  $\omega$ -continuous, Kleene’s Fixpoint Theorem (1) implies that

$$E_C^\infty = \bigsqcup_{n<\omega} E_C^n(\perp) : V \rightarrow \overline{CT_\Sigma^\perp}(O)$$

is the least fixpoint of  $E_C$  where  $\perp : V \rightarrow \overline{CT_\Sigma^\perp}(O)$  maps every  $x \in V$  to  $\Omega$ .

Hence it remains to show that every  $g : V \rightarrow \overline{CT_\Sigma}(O)$  with (1) agrees with  $E_C^\infty$ .

So let  $B = \bigcup \mathcal{I}$  and  $g : V \rightarrow \overline{CT_\Sigma}(O)$  satisfy (1). Since  $E_C^\infty$  is the least function that satisfies (1),

$$E_C^\infty \leq g. \quad (3)$$

Below we show that for all  $t \in \overline{T_\Sigma}(V + O)$  and  $n \in \mathbb{N}$ ,

$$\text{def}([g, \text{inc}_O]^*(t)) \cap B^n \subseteq \text{def}([E_C^{n+1}(\perp), \text{inc}_O]^*(t)), \quad (4)$$

in particular, for all  $x \in V$ ,

$$\text{def}(g(x)) \cap B^n \subseteq \text{def}(E_C^{n+1}(\perp)(x)). \quad (5)$$

(5) implies

$$\text{def}(g(x)) \subseteq \bigcup_{n < \omega} \text{def}(E_C^n(\perp)(x)) = \text{def}(\bigsqcup_{n < \omega} E_C^n(\perp)(x)) = \text{def}(E_C^\infty(x))$$

and thus  $g \leq E_C^\infty$ . Hence by (3),  $g = E_C^\infty$ .

*Proof of (4) by induction on n.*

Let  $t \in \overline{T_\Sigma}(V + O)$ ,  $n \in \mathbb{N}$ ,  $\mathbf{h} = [g, \text{inc}_O]$  and  $\mathbf{h}_n = [E_C^n(\perp), \text{inc}_O]$ .

*Case 1:*  $t = *$ . Then  $\text{def}([g, \text{inc}_O]^*(t)) \cap B^0 = 1$ , for all  $n > 0$ ,  $\text{def}([g, \text{inc}_O]^*(t)) \cap B^n = \emptyset$ , and for all  $n \in \mathbb{N}$ ,  $\text{def}([E_C^{n+1}(\perp), \text{inc}_O]^*(t)) = 1$ . Hence (4) holds true.

*Case 2:*  $t \in V$  and  $E(t) = d(u)$  for some  $d : s \rightarrow e \in D$  and  $u \in \overline{T_\Sigma}(V + O)_e$ .

By (2),

$$h^*(t) = g(t) = h^*(E(t)) = h^*(d(u)) = d(h^*(u)), \quad (6)$$

$$h_{n+1}^*(t) = E_C^{n+1}(\perp)(t) = E_C(E_C^n(\perp))(t) = h_n^*(E(t)) = h_n^*(d(u)) = d(h_n^*(u)). \quad (7)$$

*Case 2.1:*  $n = 0$ . Then

$$\text{def}(h^*(t)) \cap B^n = \text{def}(h^*(t)) \cap B^0 = \text{def}(h^*(t)) \cap 1 \stackrel{(6)}{=} 1 \stackrel{(7)}{\subseteq} \text{def}(h_{n+1}^*(t)).$$

*Case 2.2:*  $n > 0$ . Let  $w \in \text{def}(h^*(t)) \cap B^n$ . By (6),  $w \in \text{def}(d(h^*(u)))$  and thus  $w = bv$  for some  $b \in B$  and  $v \in \text{def}(h^*(u)) \cap B^{n-1}$ . By induction hypothesis,  $v \in \text{def}(h_n^*(u))$ .

Hence

$$w = bv \in \text{def}(d(h_n^*(u))) \stackrel{(7)}{=} \text{def}(h_{n+1}^*(t)).$$

Therefore, (4) holds true in both subcases.

*Case 3:*  $t \in O$ . Then  $\text{def}(h^*(t)) \cap B^n = \text{def}(t) \cap B^n = 1 = \text{def}(t) = \text{def}(h_{n+1}^*(t))$ .

Case 4:  $t = d(u)$  for some  $d : s \rightarrow e \in D$  and  $u \in \overline{T_\Sigma}(V + O)_e$ . Then

$$h^*(t) = h^*(d(u)) = d(h^*(u)), \quad (8)$$

$$h_{n+1}^*(t) = h_{n+1}^*(d(u)) = d(h_{n+1}^*(u)) \quad (9)$$

Case 4.1:  $n = 0$ . Then

$$\text{def}(h^*(t)) \cap B^n = \text{def}(h^*(t)) \cap B^0 = \text{def}(h^*(t)) \cap 1 \stackrel{(8)}{=} 1 \stackrel{(9)}{\subseteq} \text{def}(h_{n+1}^*(t)).$$

Case 4.2:  $n > 0$ . Let  $w \in \text{def}(h^*(t)) \cap B^n$ . By (8),  $w \in \text{def}(h^*(u))$  and thus  $w = bv$  for some  $b \in B$  and  $v \in \text{def}(h^*(u)) \cap B^{n-1}$ . By induction hypothesis,  $v \in \text{def}(h_n^*(u))$ . Since  $h_n \leq h_{n+1}$  and  $\overline{CT}_\Sigma^\perp(O) \in \text{Poset}^S$ ,  $h_n^* \leq h_{n+1}^*$ . Hence

$$w = bv \in \text{def}(d(h_n^*(u))) \subseteq \text{def}(d(h_{n+1}^*(u))) \stackrel{(9)}{=} \text{def}(h_{n+1}^*(t)).$$

Therefore, (4) holds true in both subcases.

Case 5:  $t = i(u) \in \overline{T_\Sigma}(V + O)_e$  for some  $e = \prod_{i \in I} e_i$ ,  $i \in I$  and  $u \in \overline{T_\Sigma}(V + O)_{e_i}$ . Then we obtain (4) as in Case 4 with  $i$  instead of  $d$ .

*Case 6:*  $t = ()\{i \rightarrow t_i \mid i \in I\} \in \overline{T_\Sigma}(V + O)_e$  for some  $e = \coprod_{i \in I} e_i$  and  $(t_i)_{i \in I} \in \bigtimes_{i \in I} T_\Sigma(V + O)_{e_i}$ . Then for all  $i \in I$ ,

$$\pi_i(h^*(t)) = h^*(t_i) = \pi_i((())\{i \rightarrow h^*(t_i) \mid i \in I\}),$$

$$\pi_i(h_{n+1}^*(t)) = \pi_i((())\{i \rightarrow h_{n+1}^*(t_i) \mid i \in I\}).$$

Hence

$$h^*(t) = ((())\{i \rightarrow h^*(t_i) \mid i \in I\}), \quad (10)$$

$$h_{n+1}^*(t) = ((())\{i \rightarrow h_{n+1}^*(t_i) \mid i \in I\}) \quad (11)$$

*Case 6.1:*  $n = 0$ . Then

$$\text{def}(h^*(t)) \cap B^n = \text{def}(h^*(t)) \cap B^0 = \text{def}(h^*(t)) \cap 1 \stackrel{(10)}{=} 1 \stackrel{(11)}{\subseteq} \text{def}(h_{n+1}^*(t)).$$

*Case 6.2:*  $n > 0$ . Let  $w \in \text{def}(h^*(t)) \cap B^n$ . By (10),  $w \in \text{def}((())\{i \rightarrow h^*(t_i) \mid i \in I\})$  and thus  $w = iv$  for some  $i \in I$  and  $v \in \text{def}(h^*(t_i)) \cap B^{n-1}$ . By induction hypothesis,  $v \in \text{def}(h_n^*(t_i))$ . Since  $h_n \leq h_{n+1}$  and  $\overline{CT_\Sigma^\perp}(O) \in \text{Poset}^S$ ,  $h_n^* \leq h_{n+1}^*$ . Hence

$$w = iv \in \text{def}((())\{i \rightarrow h_n^*(t_i) \mid i \in I\}) \subseteq \text{def}((())\{i \rightarrow h_{n+1}^*(t_i) \mid i \in I\}) \stackrel{(11)}{=} \text{def}(h_{n+1}^*(t)).$$

Therefore, (4) holds true in both subcases.  $\square$

## Word acceptors

Let  $X$  be a set of “input elements” and  $A$  be a finite set of “states”.

A **bottom-up  $X$ -word acceptor** is a pair  $(\mathcal{A}, B)$  that consists of a  $Dyn(X, 1)$ -algebra  $\mathcal{A}$  with carrier  $A$  and a subset  $B$  of  $A_{state}$  whose elements are called **final states**.

In chapter 9 we have seen that  $X^*$  is the carrier of the initial  $List(X)$ -algebra ([sample algebra 3](#)).

$(\mathcal{A}, B)$  accepts the language  $L(\mathcal{A}, B) =_{def} \{w \in X^* \mid \text{fold}^{\mathcal{A}}(w) \in B\}$ .

$L \subseteq X^*$  is **bottom-up regular** if there are a bottom-up  $X$ -word acceptor  $(\mathcal{A}, B)$  such that  $L(\mathcal{A}, B) = L$ .

A **deterministic (non-deterministic) top-down  $X$ -word acceptor** is a pair  $(\mathcal{A}, a)$  that consists of an  $Acc(X)$ -algebra ( $NAcc(X)$ -algebra)  $\mathcal{A}$  with carrier  $A$  and an element  $a \in A_{state}$  for some  $s \in S$  that is called an **initial state**.

In chapter 9 we have seen that  $\mathcal{P}(X^*)$  is the carrier of both the final  $Acc(X)$ -algebra  $Pow(X)$  ([sample algebra 20](#)) and the final  $NAcc(X)$ -algebra  $NPow(X)$  ([sample algebra 21](#)).

Hence the unique  $\{N\}Acc(X)$ -homomorphism  $unfold^{\mathcal{A}} : \mathcal{A} \rightarrow \{N\}Pow(X)$  maps states to subsets of  $X^*$ .

$(\mathcal{A}, a)$  accepts the language  $L(\mathcal{A}, a) =_{def} unfold^{\mathcal{A}}(a)$ .

$L \subseteq X^*$  is **regular** (or **deterministic regular**) if there are a nondeterministic (or deterministic) top-down  $X$ -word acceptor  $(\mathcal{A}, a)$  such that  $unfold^{\mathcal{A}}(a) = L$ .

Bottom-up regular languages are regular and vice versa.

Let  $Lang(X)$  be defined as sample algebra 19. The Brzozowski automaton (see sample algebra 23) provides an acceptor for every deterministic regular language:

Since  $T_{Reg(X)}$  is a  $Acc(X)$ -algebra,  $Pow(X)$  is a final one and

$$fold^{Lang(X)} : T_{Reg(X)} \rightarrow Lang(X)$$

is  $Acc(X)$ -homomorphic,

$$fold^{Lang(X)} = unfold^{Bro(X)}. \quad (1)$$

In the section Biinductive definition of regular operators, (1) is derived from the form of equations that define  $\text{Bro}(X)$ .

Regular languages are deterministic regular and vice versa.

*Proof.* “ $\Leftarrow$ ”: trivial.

“ $\Rightarrow$ ”: Let  $\mathcal{A}$  be a nondeterministic  $X$ -word acceptor and  $\mathcal{A}'$  be the deterministic  $X$ -word acceptor with carrier  $\mathcal{P}(A)$  whose operations are defined as follows:

$$\begin{aligned}\delta^{\mathcal{A}'} : \mathcal{P}(A) &\rightarrow \mathcal{P}(A)^X \\ B &\mapsto \lambda x. \bigcup_{a \in B} \delta^{\mathcal{A}}(a)(x) \\ \beta^{\mathcal{A}'} : \mathcal{P}(A) &\rightarrow 2 \\ B &\mapsto \max\{\beta^{\mathcal{A}}(a) \mid a \in B\}\end{aligned}$$

Suppose that for all  $B \subseteq A$ ,

$$\text{unfold}^{\mathcal{A}'}(B) = \bigcup_{a \in B} \text{unfold}^{\mathcal{A}}(a). \quad (2)$$

Then in particular,  $\text{unfold}^{\mathcal{A}'}(\{a\}) = \text{unfold}^{\mathcal{A}}(a)$  for all  $a \in A$ , i.e.,  $\mathcal{A}'$  and  $\mathcal{A}$  accept the same languages. (2) is equivalent to (3): For all  $w \in X^*$ ,

$$w \in \text{unfold}^{\mathcal{A}'}(B) \Leftrightarrow \exists a \in B : w \in \text{unfold}^{\mathcal{A}}(a). \quad (3)$$

*Proof of (3) by induction on  $|w|$ .*

$$\begin{aligned} \epsilon \in \text{unfold}^{\mathcal{A}'}(B) &\Leftrightarrow \max\{\beta^{\mathcal{A}}(a) \mid a \in B\} = \beta^{\mathcal{A}'}(B) = 1 \Leftrightarrow \exists a \in B : \beta^{\mathcal{A}}(a) = 1 \\ &\Leftrightarrow \exists a \in B : \epsilon \in \text{unfold}^{\mathcal{A}}(a). \end{aligned}$$

For all  $x \in X$  and  $w \in X^*$ ,

$$\begin{aligned} x \cdot w \in \text{unfold}^{\mathcal{A}'}(B) &\Leftrightarrow w \in \text{unfold}^{\mathcal{A}'}(\delta^{\mathcal{A}'}(B)(x)) \\ &\stackrel{\text{ind. hyp.}}{\Leftrightarrow} \exists b \in \delta^{\mathcal{A}'}(B)(x) = \bigcup_{a \in B} \delta^{\mathcal{A}}(a)(x) : w \in \text{unfold}^{\mathcal{A}}(b) \\ &\Leftrightarrow \exists a \in B, b \in \delta^{\mathcal{A}}(a)(x) : w \in \text{unfold}^{\mathcal{A}}(b) \Leftrightarrow \exists a \in B : x \cdot w \in \text{unfold}^{\mathcal{A}}(a). \quad \square \end{aligned}$$

(2) can also be derived from the fact that  $\mathcal{A}$  and  $\mathcal{A}'$  correspond to equivalent systems of regular equations:

Let  $A$  be a set of “language variables”.

A **system of regular (word) equations** is a function  $E : A \rightarrow T_{Reg(X)}(A)$  such that for all  $a \in A$ ,

$$E(a) = par(\dots (par(seq(x_1, a_1), seq(x_2, a_2), \dots), seq(x_n, a_n)) \quad (4)$$

or

$$E(a) = par(\dots (par(\epsilon, seq(x_1, a_1)), seq(x_2, a_2)), \dots), seq(x_n, a_n) \quad (5)$$

for some  $x_1, \dots, x_n \in X$  and  $a_1, \dots, a_n \in A$ . (4) and (5) are abbreviated as

$$E(a) = x_1 \cdot a_1 + x_2 \cdot a_2 + \dots + x_n \cdot a_n \quad (6)$$

and

$$E(a) = 1 + x_1 \cdot a_1 + x_2 \cdot a_2 + \dots + x_n \cdot a_n, \quad (7)$$

respectively.

$g : A \rightarrow \mathcal{P}(X^*)$  solves  $E$  in  $\text{Lang}(X)$  (see sample algebra 19) if  $g^* \circ E = g$ .

Let  $\mathcal{A}$  be a nondeterministic  $X$ -word acceptor with carrier  $A$  and

$$E(\mathcal{A}) : A \rightarrow T_{Reg(X)}(A)$$

be the system of regular equations that is defined as follows:

For all  $a \in A$ ,

$$E(\mathcal{A})(a) = \begin{cases} \sum\{x \cdot b \mid x \in X, b \in \delta^{\mathcal{A}}(a)(x)\} & \text{if } \beta^{\mathcal{A}}(a) = 0 \\ 1 + \sum\{x \cdot b \mid x \in X, b \in \delta^{\mathcal{A}}(a)(x)\} & \text{if } \beta^{\mathcal{A}}(a) = 1 \end{cases}$$

*unfold*<sup>mathcal{A}</sup> solves  $E(\mathcal{A})$  in  $\text{Lang}(X)$  uniquely. (8)

*Proof.* By (1),  $h =_{\text{def}} \text{unfold}^{\mathcal{A}} : A \rightarrow \mathcal{P}(X^*)$  is  $\text{Reg}(X)$ -homomorphic.

For the definition of *unfold*<sup>mathcal{A}</sup>, see [Sample final algebras](#).

Let  $a \in A$  and case (6) hold true. Then  $\beta^{\mathcal{A}}(a) = 0$ . Hence

$$\begin{aligned} h^*(E(\mathcal{A})(a)) &= h^*(\sum\{x \cdot b \mid x \in X, b \in \delta^{\mathcal{A}}(a)(x)\}) \\ &= \bigcup\{h^*(x) \cdot h^*(b) \mid x \in X, b \in \delta^{\mathcal{A}}(a)(x)\} = \bigcup\{x \cdot h(b) \mid x \in X, b \in \delta^{\mathcal{A}}(a)(x)\} \\ &= \{x \cdot w \mid x \in X, w \in h(b), b \in \delta^{\mathcal{A}}(a)(x)\} = h(a). \end{aligned}$$

Let  $a \in A$  and case (7) hold true. Then  $\beta^{\mathcal{A}}(a) = 1$ . Hence

$$\begin{aligned} h^*(E(\mathcal{A})(a)) &= h^*(1 + \sum\{x \cdot b \mid x \in X, b \in \delta^{\mathcal{A}}(a)(x)\}) \\ &= h^*(1) \cup \bigcup\{h^*(x) \cdot h^*(b) \mid x \in X, b \in \delta^{\mathcal{A}}(a)(x)\} \\ &= 1 \cup \bigcup\{x \cdot h(b) \mid x \in X, b \in \delta^{\mathcal{A}}(a)(x)\} \\ &= 1 \cup \{x \cdot w \mid x \in X, w \in h(b), b \in \delta^{\mathcal{A}}(a)(x)\} = h(a). \end{aligned}$$

Hence  $\text{unfold}^{\mathcal{A}}$  solves  $E(\mathcal{A})$ .

Uniqueness follows from the fact that  $E(\mathcal{A})$  is a system of iterative  $\text{Reg}(X)$ -equations (see chapter 14), which allows us to apply Corollary RECITER:

Let  $(S, \mathcal{I}, C) = \text{Reg}(X)$ ,

$$C_A = \{a : 1 \rightarrow \text{state} \mid a \in A\}$$

and for all  $a \in A$ ,  $op_{a,\delta} : 1 \rightarrow \text{state}^X$  and  $op_{a,\delta} : 1 \rightarrow 2$  are defined as follows:

- $E(\mathcal{A})(a) = x_1 \cdot a_1 + x_2 \cdot a_2 + \cdots + x_n \cdot a_n$  implies

$$op_{a,\delta} = \text{if } * * * *$$

Conversely, let  $E : A \rightarrow T_{\text{Reg}(X)}(A)$  be a system of regular equations and  $\mathcal{A}(E)$  be the nondeterministic  $X$ -word acceptor with carrier  $A$  whose operations are defined as follows:

$$\delta^{\mathcal{A}(E)} : A \rightarrow \mathcal{P}_\omega(A)^X$$

$$a \mapsto \lambda x. \{a_i \mid 1 \leq i \leq n, x_i = x\} \text{ if (6) or (7) holds true}$$

$$\beta^{\mathcal{A}(E)} : A \rightarrow 2$$

$$a \mapsto \begin{cases} 0 & \text{if (6) holds true} \\ 1 & \text{if (7) holds true} \end{cases}$$

$unfold^{\mathcal{A}(E)}$  solves  $E$  in  $\text{Lang}(X)$  uniquely. (9)

*Proof.* Obviously,  $E(\mathcal{A}(E)) = E$ . Hence (8) implies (9). □

## Tree acceptors

The literature on tree acceptors spans over many decades (see, e.g., [166, 141, 32, 163, 138, 42, 46]). In contrast to word acceptors, (co)algebraic approaches avoiding grammars or other rewrite systems are still rare. In the following, we give one that captures the basic notions of [42], section 1.6, and [163], section 3.2.2.

The tree language to be accepted is a set of  $\Sigma$ -terms where  $\Sigma = (S, \mathcal{I}, F)$  is a finitary signature (see chapter 8).

A **bottom-up  $\Sigma$ -term acceptor** is a pair  $(\mathcal{A}, B)$  that consists of a  $\Sigma$ -algebra  $\mathcal{A}$  and an  $S$ -sorted subset  $B$  of the carrier of  $\mathcal{A}$  whose elements are called **final states**.

Above we have seen that  $T_\Sigma$  is the carrier of an initial  $\Sigma$ -algebra.

$(\mathcal{A}, B)$  **accepts** the language  $L(\mathcal{A}, B) =_{def} \{t \in T_\Sigma \mid fold^{\mathcal{A}}(t) \in B\}$ .

A **deterministic (non-deterministic) top-down  $\Sigma$ -term acceptor** is a pair  $(\mathcal{A}, a)$  that consists of a  $T\text{Acc}(\Sigma)$ -algebra ( $NT\text{Acc}(\Sigma)$ -algebra)  $\mathcal{A}$  and an element of the carrier of  $\mathcal{A}$  that is called an **initial state**.

In chapter 9 we have seen that  $\mathcal{P}(T_\Sigma)$  is the carrier of both the final  $T\text{Acc}(\Sigma)$ -algebra  $TPow(\Sigma)$  (sample algebra 29) and the final  $NT\text{Acc}(\Sigma)$ -algebra  $NTPow(\Sigma)$  (sample algebra 30). Hence the unique  $\{N\} T\text{Acc}(\Sigma)$ -homomorphism  $unfold^{\mathcal{A}} : \mathcal{A} \rightarrow \{N\} T\text{Pow}(\Sigma)$  maps states to subsets of  $T_\Sigma$ .

$(\mathcal{A}, a)$  **accepts** the language  $L(\mathcal{A}, a) =_{def} unfold^{\mathcal{A}}(a)$ .

$L \subseteq T_\Sigma$  is **regular** if there is a bottom-up acceptor  $(\mathcal{A}, B)$  or, equivalently, a non-deterministic top-down acceptor  $(\mathcal{A}, a)$  such that  $L(\mathcal{A}, B) = L$  and  $L(\mathcal{A}, a) = L$ , respectively, and the carrier of  $\mathcal{A}$  is finite.

$L \subseteq T_\Sigma$  is **deterministic top-down regular** if there is a deterministic top-down acceptor  $(\mathcal{A}, a)$  of trees such that  $L(\mathcal{A}, a) = L$  and the carrier of  $\mathcal{A}$  is finite.

Given  $L \subseteq T_\Sigma$ , the **path closure** of  $L$ ,  $\text{cl}(L)$ , is the least subset  $T$  of  $T_\Sigma$  that contains  $L$  and satisfies the following implication: for all  $c : s_1 \times \dots \times s_n \rightarrow s \in C$  and  $1 \leq i < j \leq n$ ,

$$\begin{aligned} c(t_1, \dots, t_{i-1}, u_i, t_{i+1}, \dots, t_n), c(t_1, \dots, t_{j-1}, u_j, t_{j+1}, \dots, t_n) \in T \\ \Rightarrow c(t_1, \dots, t_{i-1}, u_i, t_{i+1}, \dots, t_{j-1}, u_j, t_{j+1}, \dots, t_n) \in T. \end{aligned}$$

Previous definitions of path closure can be found in [44], section 4, and [42], section 1.8.  $L \subseteq T_\Sigma$  is **path closed** if  $L = \text{cl}(L)$ .

$L \subseteq T_\Sigma$  is deterministic top-down regular iff  $L$  is regular and path-closed. \*\*\*\*

*Proof.* “ $\Rightarrow$ ”: Let  $(\mathcal{A}, a)$  be a deterministic top-down tree acceptor with  $\text{unfold}^{\mathcal{A}}(a) = L$  and

$$c(t_1, \dots, t_{i-1}, u_i, t_{i+1}, \dots, t_n), c(t_1, \dots, t_{j-1}, u_j, t_{j+1}, \dots, t_n) \in L.$$

By the definition of  $\text{unfold}^{\mathcal{A}}$  (see above), there are  $a_1, \dots, a_n$  in the carrier of  $\mathcal{A}$  such that  $\delta_c(a) = (a_1, \dots, a_n)$ ,  $u_i \in \text{unfold}^{\mathcal{A}}(a_i)$ ,  $u_j \in \text{unfold}^{\mathcal{A}}(a_i)$  and for all  $1 \leq i \leq n$ ,  $t_i \in \text{unfold}^{\mathcal{A}}(a_i)$ . Hence  $c(t_1, \dots, t_{i-1}, u_i, t_{i+1}, \dots, t_{j-1}, u_j, t_{j+1}, \dots, t_n) \in \text{unfold}^{\mathcal{A}}(a) = L$ . Therefore,  $L$  is path-closed.

“ $\Leftarrow$ ”: Given an  $NTAcc(\Sigma)$ -algebra  $\mathcal{A}$  with carrier  $A$ , there is  $TAcc(\Sigma)$ -algebra  $\mathcal{A}'$  with carrier  $\mathcal{P}(A)$  such that for all  $B \subseteq A$ ,

$$L(\mathcal{A}', B) = \bigcup_{a \in B} cl(L(\mathcal{A}, a)), \quad (1)$$

or, equivalently, for all  $t \in T_\Sigma$ ,

$$t \in unfold^{\mathcal{A}'}(B) \iff \exists a \in B : t \in cl(unfold^{\mathcal{A}}(a)). \quad (2)$$

The operations of  $\mathcal{A}'$  are defined as follows: For all  $c : s_1 \times \cdots \times s_n \rightarrow s \in C$ ,

$$\begin{aligned} \delta_c^{\mathcal{A}'} : \mathcal{P}(A) &\rightarrow \mathcal{P}(A)^n \\ B &\mapsto \bigcup_{a \in B} \delta_c^{\mathcal{A}}(a) \end{aligned}$$

*Proof of (2) by induction on the number  $n$  of  $C$ -occurrences in  $t$ .*

*Case 1.*  $t = c(a_1, \dots, a_n)$  for some  $c : A_1 \times \cdots \times A_n \rightarrow s$ ,  $A_1, \dots, A_n \in \mathcal{I}$  and  $a_i \in A_i$  for all  $1 \leq i \leq n$ . Hence for all  $B \subseteq A$ ,  $\bigcup_{a \in B} \delta_c^{\mathcal{A}}(a) = \delta_c^{\mathcal{A}'}(B) = (B_1, \dots, B_n)$  implies

$$t \in unfold^{\mathcal{A}'}(B) \iff \forall 1 \leq i \leq n : a_i \in unfold^{\mathcal{A}'}(B_i) = B_i.$$

[44], Theorem 5, characterizes deterministic top-down regular languages as the *path closed* ones. The notions and lemmas leading to this result are as follows.

Given  $t \in T_{C\Sigma}$ , the **path language** of  $t$ ,  $\text{paths}(t)$ , is the **least** set of nonempty lists over  $X = (C' \times \mathbb{N}) \cup C''$  such that for all  $c \in C'$ ,  $c' \in C''$ ,  $i > 0$  and  $p \in X^*$ ,

- if  $t(\epsilon) = c$  and  $p \in \text{paths}(\lambda w.t(iw))$ , then  $(c, i) : p \in \text{paths}(t)$ ,
- if  $t(\epsilon) = c'$ , then  $\text{paths}(t) = \{c'\}$ .

Given  $L \subseteq T_{C\Sigma}$ ,

$$\begin{aligned}\text{paths}(L) &=_{\text{def}} \bigcup \{\text{paths}(t) \mid t \in L\}, \\ \text{pclosure}(L) &=_{\text{def}} \{t \in T_{C\Sigma} \mid \text{paths}(t) \subseteq \text{paths}(L)\}\end{aligned}$$

are called the **path language** and **path closure** of  $L$ , respectively.  $L$  is **path closed** if  $L = \text{pclosure}(L)$ .

- (1) If  $L \subseteq T_{C\Sigma}$  is top-down regular, then  $\text{paths}(L)$  is regular.
- (2) If  $L \subseteq T_{C\Sigma}$  is top-down regular, then  $\text{pclosure}(L)$  is deterministic top-down regular.
- (3) If  $L \subseteq T_{C\Sigma}$  is deterministic top-down regular, then  $L$  is path closed.

The converse of (3) immediately follows from (2).

**Example** Let  $C = \{f : state^2 \rightarrow state, c, d : 1 \rightarrow state\}$ . The language

$$L =_{def} \{f(c, d), f(d, c)\} \subseteq T_{C\Sigma}$$

(mentioned in [42], Prop. 1.6.1, [163], Remark 3.35, and [46], Thm. 10) is not path closed because  $f(c, c) \notin L$ , although

$$\begin{aligned} paths(f(c, c)) &= \{[(f, 1), c], [(f, 2), c]\} \subseteq \{[(f, 1), c], [(f, 2), d], [(f, 1), d], [(f, 2), c]\} \\ &= paths(L). \end{aligned}$$

It is also easy to see that for every initial automaton  $(\mathcal{A}, a)$  with carrier  $A$  is a  $TAcc(C)$ -algebra,

$$L \subseteq unfold^{\mathcal{A}}(a) \text{ implies } f(c, c) \in unfold^{\mathcal{A}}(a) :$$

Let  $L \subseteq unfold^{\mathcal{A}}(a)$ . Then there are  $b, b' \in A$  such that  $\delta_f^{\mathcal{A}}(a) = (b, b')$ ,  $\beta_c^{\mathcal{A}}(b) = 1$  and  $\beta_d^{\mathcal{A}}(b') = 1$  (because  $f(c, d) \in unfold^{\mathcal{A}}(a)$ ), but also  $\beta_d^{\mathcal{A}}(b) = 1$  and  $\beta_c^{\mathcal{A}}(b') = 1$  (because  $f(d, c) \in unfold^{\mathcal{A}}(a)$ ).  $\beta_c^{\mathcal{A}}(b) = 1 = \beta_c^{\mathcal{A}}(b')$  implies  $f(c, c) \in unfold^{\mathcal{A}}(a)$ .  $\square$

Path language for trees with infinite paths: Given  $t \in CT_{C\Sigma}$ , the **path language** of  $t$ ,  $\text{paths}(t)$ , is the greatest set of nonempty colists over  $X = (C' \times \mathbb{N}) \cup C''$  such that for all  $c \in C'$ ,  $c' \in C''$ ,  $i > 0$  and  $p \in X^*$ ,

- $(c, i) : p \in \text{paths}(t)$  implies  $t(\epsilon) = c$ ,  $t(i) \in \text{src}(t)$  and  $p \in \text{paths}(\lambda w.t(iw))$ ,
- $c' : p \in \text{paths}(t)$  implies  $\text{def}(t) = 1$ ,  $t(\epsilon) = c'$  and  $p = \epsilon$ .

Let  $L \subseteq T_\Sigma$ ,  $\sim_L$  be the **Nerode relation of  $L$** , i.e., the greatest  $\Sigma$ -congruence that is contained in the kernel of  $\text{set2fun}(L) : T_\Sigma \rightarrow 2$ ,  $Q = \{[t]_{\sim_L} \mid t \in L\}$  and  $\mathcal{F}(L) =_{\text{def}} (T_\Sigma/\sim_L, Q)$ . Hence

$$\begin{aligned} L(\mathcal{F}(L)) &= \{t \in T_\Sigma \mid \text{fold}^{T_\Sigma/\sim_L}(t) \in Q\} = \{t \in T_\Sigma \mid \text{nat}_{\sim_L}(t) \in Q\} \\ &= \{t \in T_\Sigma \mid t \in L\} = L \end{aligned}$$

and thus  $\mathcal{F}(L)$  is a bottom-up tree acceptor of  $L$ .

Sets of terms that represent XML documents satisfying certain constraints can often be described as regular term languages. Other formalizations of XML constraints use second-order or modal logics (see, e.g., chapter 27).

## Top-down tree automata in Kleisli categories

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature. A **nondeterministic  $\Sigma$ -algebra  $\mathcal{A}$**  consists of an  $S$ -sorted set  $A$  and a function  $f^{\mathcal{A}} : A_e \rightarrow \mathcal{P}(A_{e'})$  for every  $f : e \rightarrow e' \in F$ .

Let  $\mathcal{A}, \mathcal{B}$  be nondeterministic  $\Sigma$ -algebra with carriers  $A$  and  $B$ , respectively. A multivalued  $S$ -sorted function  $h : A \rightarrow B$  is a **multivalued  $\Sigma$ -homomorphism** from  $\mathcal{A}$  to  $\mathcal{B}$  if for all  $f : e \rightarrow e' \in F$ ,

$$h_{e'} \circ_{\mathcal{P}} f^{\mathcal{A}} = f^{\mathcal{B}} \circ_{\mathcal{P}} h_e$$

where  $\circ_{\mathcal{P}}$  is the Kleisli composition of the powerset monad  $\mathcal{P}$  (see chapter 23) and  $h_e$  and  $h_{e'}$  are instances of a lifting of  $h$  to a *multivalued*  $\mathcal{T}_p(S, \mathcal{I})$ -sorted function.

$ndAlg_{\Sigma}$  denotes the subcategory of  $Set_{\mathcal{P}}^S$  (see chapter 7) that consists of all nondeterministic  $\Sigma$ -algebras and multivalued  $\Sigma$ -homomorphisms.

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive signature,  $(S', D) = co\Sigma$  (see chapter 13). A nondeterministic  $co\Sigma$ -algebra  $\mathcal{L}(\Sigma)$  of tree languages may be defined as follows:

For all  $s \in S'$ ,

$$\begin{aligned}
 \mathcal{L}(\Sigma)_s &= T_{\Sigma,s}, \\
 d_s^{\mathcal{L}(\Sigma)} : T_{\Sigma,s} &\rightarrow \mathcal{P}(\coprod_{c:e \rightarrow s \in C} T_{\Sigma,e}) \\
 c\{i \rightarrow t_i \mid i \in I\} &\mapsto \{((t_i)_{i \in I}, c)\}.
 \end{aligned}$$

Provided that results of [80], section 5; [79], section 3.2; [66], section 3, can be adopted here, there is a distributive law

$$(\coprod_{c:e \rightarrow s \in C} \_) \circ \mathcal{P} \rightarrow \mathcal{P} \circ \coprod_{c:e \rightarrow s \in C} \_$$

and thus  $\mathcal{L}(\Sigma)$  is final in  $ndAlg_{co\Sigma}$  such that for all nondeterministic  $co\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$  and  $s \in S'$ ,

$$\begin{aligned}
 unfold_s^{\mathcal{A}} : A_s &\rightarrow \mathcal{P}(T_{\Sigma,s}) \\
 a &\mapsto (\bigsqcup_{n \in \mathbb{N}} \Phi^n(\lambda x. \emptyset))(a, s) \quad (\text{see chapter 3}) \\
 \Phi : \mathcal{P}(T_{\Sigma}) \coprod_{s \in S'} A_s &\rightarrow \mathcal{P}(T_{\Sigma}) \coprod_{s \in S'} A_s \\
 f &\mapsto \lambda(a, s). f(a, s) \cup \{c\{i \rightarrow t_i \mid i \in I\} \\
 &\quad \mid (b, c : \prod_{i \in I} s_i \rightarrow s) \in d_s^{\mathcal{A}}(a), \\
 &\quad t_i \in f(\pi_i(b), s_i) \vee (s_i \in Set_{\neq \emptyset} \wedge t_i \in s_i)\}.
 \end{aligned}$$

## Iterative equations of a CFG

Let  $G = (S, X, R)$  be a CFG (see chapter 9).

$G$  induces an iterative system of  $Reg(X)$ -equations:

$$\begin{aligned} E_G : S &\rightarrow T_{Reg(X)}(S) \\ s &\mapsto \text{par}(\dots(\text{par}(\overline{w_1}, \overline{w_2}), \dots), \overline{w_n}) \end{aligned}$$

where  $\{w_1, \dots, w_n\} = \{w \in (S \cup \mathcal{P}(X))^* \mid s \rightarrow w \in R\}$  and for all  $n > 0$ ,  $s_1, \dots, s_n \in S \cup \mathcal{P}(X)$ ,  $s \in S$  and  $L \subseteq X$ ,

$$\begin{aligned} \overline{s_1 \dots s_n} &= \text{seq}(\dots(\text{seq}(\overline{s_1}, \overline{s_2}), \dots), \overline{s_n}), \\ \overline{s} &= s. \end{aligned}$$

From now on,  $Reg(X)$ -terms of the form

$$\text{par}(\dots(\text{par}(t_1, t_2), \dots), t_n) \text{ and } \text{seq}(\dots(\text{seq}(t_1, t_2), \dots), t_n)$$

are written as  $t_1 + \dots + t_n$  and  $t_1 \cdot \dots \cdot t_n$ , respectively.

## Theorem LANG

- (i) The function  $\text{lang}_G : S \rightarrow \mathcal{P}(X^*)$  with  $\text{lang}_G(s) = L(G)_s$  for all  $s \in S$  solves  $E_G$  in  $\text{Lang}(X)$ .

Let  $g : S \rightarrow \mathcal{P}(X^*)$  solve  $E_G$  in  $\text{Lang}(X)$ .

- (ii) The  $S$ -sorted set  $\text{Sol}$  mit  $\text{Sol}_s = g(s)$  for all  $s \in S$  is the carrier of a  $\Sigma(G)$ -subalgebra of  $\text{Word}(G)$ .
- (iii) For all  $s \in S$ ,  $\text{lang}_G(s) \subseteq g(s)$ , in other words:  $L(G)$  is the least solution of  $E_G$  in  $\text{Lang}(X)$ .

*Proof.* See [125], Satz 16.3. □

Let  $G$  be **non-left-recursive**, i.e., the usual derivation relation  $\xrightarrow{+}_G$  induced by  $G$  does not contain pairs of the form  $(s, sw)$  with  $s \in S$ . Then there is  $E'_G : S \rightarrow T_{\text{Reg}(X)}(S)$  such that for all  $s \in S$ ,  $\text{Lang}(X)$  satisfies  $E'_G(s) = E_G(s)$  and there are  $n_s \in \mathbb{N}$ ,  $B_{s,1}, \dots, B_{s,n_s} \subseteq X$  and  $\text{Reg}(X)$ -terms  $t_{s,1}, \dots, t_{s,n_s}$  over  $S$  with

$$E'_G(s) = B_{s,1} \cdot t_{s,1} + \cdots + B_{s,n_s} \cdot t_{s,n_s} \quad (1)$$

or

$$E'_G(s) = 1 + B_{s,1} \cdot t_{s,1} + \cdots + B_{s,n_s} \cdot t_{s,n_s}. \quad (2)$$

Let  $Reg(X)(S)$  be the extension of  $Reg(X)$  by a constructor  $s : 1 \rightarrow reg$  for all  $s \in S$ ,

$$\begin{aligned} D\Sigma = & (\{state\}, \{2, X\}, \\ & \{max, * : 2 \times 2 \rightarrow 2\} \cup \{\_ \in B : X \rightarrow 2 \mid B \subseteq X\}, \\ & \{\delta : state \rightarrow state^X, \beta : state \rightarrow 2\}), \end{aligned}$$

\*\*\*\*  $\Psi = (Reg(X), D\Sigma)$ ,  $B \subseteq X$  and  $t, u \in V$ .

Let  $E' = \bigcup_{s \in S} E_s$  where  $E_s$  consists of the following two  $(Reg(X)(S) \cup D\Sigma)$ -equations with  $b = 0$  in case (1) and  $b = 1$  in case (2), respectively:

$$\begin{aligned} \delta(s) &= \lambda x. par(ite(x \in B_{s,1}, t_{s,1}, \bar{\emptyset}), \dots, ite(x \in B_{n_s}, t_{n_s}, \bar{\emptyset})), \\ \beta(s) &= b. \end{aligned}$$

## Theorem REGSOL

Let  $\Psi_S = (\text{Reg}(X)(S), D\Sigma)$ ,  $\Sigma = \text{Reg}(X)(S) \cup D\Sigma$ ,  $\text{sol} : S \rightarrow \text{Lang}(X)$  be a solution of  $E_G$  in  $\text{Lang}(X)$  and  $A$  be the  $\Sigma$ -algebra such that the  $\text{Reg}(X)$ - and  $\text{Acc}(X)$ -reducts of  $A$  agree with  $\text{Lang}(X)$  and  $\text{Pow}(X)$ , respectively, and for all  $s \in S$ ,  $s^A = \text{sol}(s)$ .

Let  $BRE$  be the set of Biinductive definition of regular operators.

$A$  is a coinductive solution of the system  $BRE \cup E'$  of recursive  $\Psi_S$ -equations in the final  $\text{Acc}(X)$ -algebra  $\text{Pow}(X)$ .

*Proof.* See [125], Satz 16.7. □

The  $\text{Acc}(X)$ -algebra  $\text{Bro}(X)(S)$  is defined inductively as follows:

$$\text{Bro}(X)(S)_{\text{state}} = T_{\text{Reg}(X)}(S)_{\text{state}} (= T_{\text{Reg}(X)(S), \text{state}})$$

and for all  $s \in S$ ,  $B \subseteq X$  and  $t, u \in T_{\text{Reg}(X)}(S)_{\text{state}}$ ,

$$\begin{aligned} \delta^{\text{Bro}}(s) &= \lambda x. \text{par}(\text{ite}(x \in B_{s,1}, t_{s,1}, \emptyset), \dots, \text{ite}(x \in B_{n_s}, t_{n_s}, \emptyset)), \\ \delta^{\text{Bro}}(\text{par}(t, u)) &= \lambda x. \text{par}(\delta^{\text{Bro}}(t)(x), \delta^{\text{Bro}}(u)(x)), \end{aligned}$$

$$\begin{aligned}
\delta^{Bro}(seq(t, u)) &= \lambda x.par(seq(\delta^{Bro}(t)(x), u), \\
&\quad \text{if } \beta^{Bro}(t) = 1 \text{ then } \delta^{Bro}(u)(x) \text{ else } \bar{\emptyset}), \\
\delta^{Bro}(iter(t)) &= \lambda x.seq(\delta^{Bro}(t)(x), iter(t)), \\
\delta^{Bro}(eps) &= \bar{\emptyset}, \\
\delta^{Bro}(base(B)) &= \lambda x.ite(x \in B, eps, \bar{\emptyset}), \\
\beta^{Bro}(s) &= \begin{cases} 0 & \text{in case (1),} \\ 1 & \text{in case (2),} \end{cases} \\
\beta^{Bro}(par(t, u)) &= max\{\beta^{Bro}(t), \beta^{Bro}(u)\}, \\
\beta^{Bro}(seq(t, u)) &= \beta^{Bro}(t) * \beta^{Bro}(u), \\
\beta^{Bro}(iter(t)) &= 1, \\
\beta^{Bro}(eps) &= 1, \\
\beta^{Bro}(base(B)) &= 0.
\end{aligned}$$

For  $C\Sigma = Reg(X)(S)$ ,  $Bro(X)(S)$  interprets  $D\Sigma$  the same as  $T_{C\Sigma}$  (see the proof of Theorem COINDSOL).

## Lemma SOLHOM

Let  $A$  be the coinductive solution of  $BRE \cup E'$  in  $Pow(X)$ , given by Theorem REGSOL, and  $g : S \rightarrow A$ .

- (i) If  $g$  solves  $E_G$  in  $A$ , then  $g^* : T_{Reg(X)}(S) = Bro(X)(S) \rightarrow A$  is  $Acc(X)$ -homomorphic.
- (ii) If  $g^*$  is  $Acc(X)$ -homomorphic, then  $g$  solves  $E'_G$  in  $A$ .

*Proof of (i).* Suppose that  $g$  solves  $E_G$  in  $A$ . By Lemma FRINI,

$$g^* = fold^A : T_{Reg(X)}(S) = T_{Reg(X)(S)} \rightarrow A. \quad (3)$$

Since  $Bro(X)(S)$  interprets  $D\Sigma$  the same as  $T_{Reg(X)(S)}$ , Theorem COINDSOL implies

$$unfold^{Bro(X)(S)} = fold^A : Bro(X)(S) = T_{Reg(X)(S)} \rightarrow A. \quad (4)$$

Hence for all arrows  $f$  of  $Acc(X)$ ,

$$\begin{aligned} f^A \circ g^* &\stackrel{(3)}{=} f^A \circ fold^A \stackrel{(4)}{=} f^A \circ unfold^{Bro} \circ unfold^{Bro}^{Acc(X)-hom.} \\ &\stackrel{(4)}{=} fold^A \circ f^{Bro} \stackrel{(3)}{=} g^* \circ f^{Bro}, \end{aligned}$$

i.e.,  $g^*$  is  $Acc(X)$ -homomorphic.

*Proof of (ii).* Suppose that  $g^*$  is  $Acc(X)$ -homomorphic. Let  $s \in S$ . We obtain

$$\begin{aligned} & \delta^A(g^*(E'_G(s))) \\ & \stackrel{\text{see proof of Thm. REGSOL}}{=} g^*(\lambda x.\text{par}(\text{ite}(x \in B_{s,1}, t_{s,1}, \emptyset), \dots, \text{ite}(x \in B_{s,n_s}, t_{s,n_s}, \emptyset))) \\ & = g^*(\delta^{Bro}(s)) \stackrel{g^* \text{ Acc}(X)-hom.}{=} \delta^A(g^*(s)) = \delta^A(g(s)). \end{aligned}$$

Moreover,

$$\beta^A(g^*(E'_G(s))) \stackrel{\text{see proof of Thm. REGSOL}}{=} 1 = g^*(\beta^{Bro}(s)) \stackrel{g^* \text{ Acc}(X)-hom.}{=} \beta^A(g^*(s)) = \beta^A(g(s))$$

in case (1) and

$$\beta^A(g^*(E'_G(s))) \stackrel{\text{see proof of Thm. REGSOL}}{=} 0 = g^*(\beta^{Bro}(s)) \stackrel{g^* \text{ Acc}(X)-hom.}{=} \beta^A(g^*(s)) = \beta^A(g(s))$$

in case (2). Hence  $\sim =_{def} \{(g^*(E'_G(s)), g(s)) \mid s \in S\}$  is a  $Acc(X)$ -congruence on  $A$ .

Since  $A$  is final in  $Alg_{Acc(X)}$ , Lemma FIN (3) implies that  $\Delta_A$  is the only  $co\Sigma$ -congruence on  $A$ .

We conclude  $\sim = \Delta_A$  and thus  $g^* \circ E'_G = g$ , i.e.,  $g$  solves  $E'_G$  in  $A$ . □

### Invariants and algebraic induction

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature,  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $B$  be a  $\Sigma$ -invariant of  $\mathcal{A}$ .

$inc_B : B \rightarrow A$  denotes the  $S$ -sorted **inclusion mapping** that sends every element of  $B$  to itself.

$inc_B$  is a monomorphism in  $Alg_\Sigma$  from  $\mathcal{A}|_B$  to  $\mathcal{A}$  (see chapter 9).

For all  $f : e \rightarrow e' \in Arr_\Sigma$  and  $a \in B_e$ ,

$$f^{\mathcal{A}|_B}(a) = inc_B(f^{\mathcal{A}|_B}(a)) = f^{\mathcal{A}}(inc_B(a)) = f^{\mathcal{A}}(a).$$

Let  $h : A \rightarrow B$  be an  $S$ -sorted function.

The  $S$ -sorted subset  $img(h) =_{def} \{h(a) \mid a \in A\}$  of  $B$  is called the **image of  $h$** .

$h$  is surjective iff  $img(h) = B$ .

## Lemma IMG

- (1) Let  $h : A \rightarrow B$  be an  $S$ -sorted function and  $\mathcal{B}$  be a  $\Sigma$ -algebra with carrier  $B$ .  $A$  can be extended to a  $\Sigma$ -algebra  $\mathcal{A}$  and  $h$  to a  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $\mathcal{B}$  iff  $img(h)$  is a  $\Sigma$ -invariant.
- (2)  $h : \mathcal{A} \rightarrow \mathcal{B}$  is  $\Sigma$ -homomorphic iff there is a unique  $\Sigma$ -epimorphism  $h' : \mathcal{A} \rightarrow \mathcal{B}|_{img(h)}$  with  $inc_{img(h)} \circ h' = h$ . Hence, if  $h$  is mono, then by Lemma EPIMON (2),  $h'$  is mono and thus  $\mathcal{A}$  and  $\mathcal{B}|_{img(h)}$  are  $\Sigma$ -isomorphic.

*Proof.* (1) If  $h$  is  $\Sigma$ -homomorphic, then  $img(h)$  is a  $\Sigma$ -invariant. Let  $img(h)$  be a  $\Sigma$ -invariant. For all  $f : e \rightarrow e' \in F$ , define  $f^{\mathcal{A}} : A_e \rightarrow A_{e'}$  such that for all  $a \in A_e$ ,  $f^{\mathcal{A}}(a) \in h^{-1}(f^{\mathcal{B}}(h(a)))$ , and for all  $p \in P$ , define  $p^{\mathcal{A}} = \{a \in A \mid h(a) \in p^{\mathcal{B}}\}$ . Then  $\mathcal{A}$  is a  $\Sigma$ -algebra and  $h$  is  $\Sigma$ -homomorphic.

(2)  $h'$  with  $h'(a) = h(a)$  for all  $a \in A$  has all desired properties. The uniqueness and the homomorphism property of  $h'$  follow from Lemma EMH (2).  $\square$

Moreover, by Kleene's Fixpoint Theorem (1), for every  $S$ -sorted subset  $B$  of  $A$ , the least  $\Sigma$ -invariant  $\langle B \rangle$  containing  $B$  is the union of all  $B_n$ ,  $n \in \mathbb{N}$ , with  $B_0 = B$  and for all  $s \in S$ ,

$$B_{n+1,s} = \{c^{\mathcal{A}}(a) \mid c : e \rightarrow s \in C, a \in B_{n,e}\}.$$

## Proofs by algebraic induction

- ❖ Let  $C \subseteq F$  be a set of constructors and  $C\Sigma = (S, \mathcal{I}, C)$ . A  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  **satisfies the (algebraic) induction principle for  $C$**  if for all  $S$ -sorted subsets  $B$  of  $A$ ,  $A = B$  iff  $B$  contains a  $C\Sigma$ -invariant of  $\mathcal{A}$ .

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  that satisfies the induction principle for  $C$  and for all  $s \in S$ ,  $\varphi_s$  be a  $\Sigma$ -formula such that  $\text{free}(\varphi_s) \subseteq \{x\} \subseteq V_s$ .

Then the (in)validity of  $\varphi_s$ ,  $s \in S$ , in  $\mathcal{A}$  may be proved by the following iterative algorithm:

- **Step 1:** For all  $s \in S$ , set  $B_s := B_{0,s} =_{\text{def}} \{g(x) \mid g \in \varphi_s^{\mathcal{A}}\}$ .
- **Step 2:** For all  $s \in S$ , let  $B'_s = \{c^{\mathcal{A}}(a) \mid c : e \rightarrow s \in C, a \in B_e\}$ .
- **Step 3:** If  $B' \subseteq B$ , then stop:  $B$  is a  $C\Sigma$ -invariant of contained in  $B_0$ , and thus by the induction principle for  $C$ , for all  $s \in S$ ,

$$A_s = B_{0,s} = \{g(x) \mid g \in \varphi_s^{\mathcal{A}}\}.$$

Hence for all  $s \in S$ ,  $A^V = \varphi_s^{\mathcal{A}}$ , i.e.,  $\mathcal{A}$  satisfies  $\varphi$ .

If  $B' \not\subseteq B$ , then for all  $s \in S$ , set  $B_s := B_s \cap B'_s$  and go to Step 2.

**Lemma INV**

Let  $h : \mathcal{A} \rightarrow \mathcal{B}$  be  $\Sigma$ -homomorphic,  $A$  be the carrier of  $\mathcal{A}$  and  $inv$  be a  $\Sigma$ -invariant of  $\mathcal{B}$ .

$$h^{-1}(inv) = \{a \in A \mid h(a) \in inv\}$$

is a  $\Sigma$ -invariant of  $\mathcal{A}$ .

*Proof.* Let  $f : e \rightarrow e' \in F$ ,  $a \in A_e$  and  $a \in h^{-1}(inv)$ . Then  $h(a) \in inv$  and thus  $h(f^{\mathcal{A}}(a)) = f^{\mathcal{B}}(h(a)) \in inv$  because  $h$  is  $\Sigma$ -homomorphic and  $inv$  is a  $\Sigma$ -invariant. Hence  $f^{\mathcal{A}}(a) \in h^{-1}(inv)$ . □

**Lemma INI** (Initial algebras satisfy the induction principle for their constructors)

Let  $\Sigma = (S, \mathcal{I}, F)$  be a **constructive** signature and  $\mathcal{A}, \mathcal{B}$  be  $\Sigma$ -algebras with carriers  $A, B$  and  $\mathcal{K}$  be a full subcategory of  $Alg_{\Sigma}$  that is closed under subalgebras.

- (1)  $\mathcal{A}$  satisfies the induction principle iff  $A$  is the only  $\Sigma$ -invariant of  $\mathcal{A}$ .
- (2) If  $A$  is the only  $\Sigma$ -invariant of  $\mathcal{A}$ , then all  $\Sigma$ -homomorphisms from  $\mathcal{A}$  to  $\mathcal{B}$  coincide.
- (3)  $\mathcal{A}$  is initial in  $\mathcal{K}$  iff  $A$  is the only  $\Sigma$ -invariant of  $\mathcal{A}$  and for all  $\mathcal{B} \in \mathcal{K}$  there is a  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $\mathcal{B}$ .

(4) If  $\mathcal{A}$  is initial in  $\mathcal{K}$ , then the image of the unique  $\Sigma$ -homomorphism  $\text{fold}^{\mathcal{B}} : \mathcal{A} \rightarrow \mathcal{B}$  is the least  $\Sigma$ -invariant of  $\mathcal{B}$ .

*Proof.*

(1) “ $\Rightarrow$ ”: Every  $\Sigma$ -invariant  $B$  of  $\mathcal{A}$  is contained in  $B$ . Hence  $B$  agrees with  $A$  if  $\mathcal{A}$  satisfies the induction principle.

“ $\Leftarrow$ ”: Suppose that  $B \subseteq A$  contains a  $\Sigma$ -invariant and  $A$  is the only  $\Sigma$ -invariant of  $\mathcal{A}$ . Then  $A \subseteq B$ .

(2) Let  $g, h : \mathcal{A} \rightarrow \mathcal{B}$  be  $\Sigma$ -homomorphisms. Then  $B = \{a \in A \mid g(a) = h(a)\}$  is a  $\Sigma$ -invariant of  $\mathcal{A}$ : Let  $f : e \rightarrow e' \in F$  and  $a \in A_e$ .

Since  $g$  and  $h$  are  $\Sigma$ -homomorphic,  $g_{e'}(f^{\mathcal{A}}(a)) = f^{\mathcal{B}}(g_e(a)) = f^{\mathcal{B}}(h_e(a)) = h_{e'}(f^{\mathcal{A}}(a))$ . Since  $g$  and  $h$  are  $S$ -sorted, Lemma LIFT (3) implies  $f^{\mathcal{A}}(a) \in B_{e'}$ . Since  $A$  is the only  $\Sigma$ -invariant of  $\mathcal{A}$ ,  $B$  agrees with  $A$  and thus for all  $a \in A$ ,  $g(a) = h(a)$ . Hence  $g = h$ .

(3) “ $\Rightarrow$ ”: Let  $\mathcal{A}$  be initial in  $\mathcal{K}$  and  $B$  be a  $\Sigma$ -invariant of  $A$ .  $B$  induces a  $\Sigma$ -monomorphism  $\text{inc}_B : \mathcal{A}|_B \rightarrow \mathcal{A}$ . Hence Lemma MINMAX (1) implies that  $\text{inc}_B$  is iso in  $\mathcal{K}$  and thus  $B = A$ . Since  $\mathcal{A}$  is initial in  $\mathcal{K}$ , there is a  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $\mathcal{B}$ .

“ $\Leftarrow$ ”: Suppose that  $A$  is the only  $\Sigma$ -invariant of  $\mathcal{A}$  and for all  $\mathcal{B} \in \mathcal{K}$  there is a  $\Sigma$ -homomorphism  $h : \mathcal{A} \rightarrow \mathcal{B}$ . By (2),  $h$  is unique. Hence  $\mathcal{A}$  is initial in  $\mathcal{K}$ .

(4) Let  $inv$  be a  $\Sigma$ -invariant of  $\mathcal{B}$ . Since  $\mathcal{A}$  is initial in  $\mathcal{K}$ , the following diagram commutes:

$$\begin{array}{ccc}
 \mathcal{A} & \xrightarrow{\text{fold}^{\mathcal{B}}} & \mathcal{B} \\
 & \searrow \text{fold}^{\mathcal{B}|_{inv}} & \nearrow \text{inc}_{inv} \\
 & \mathcal{B}|_{inv} &
 \end{array}$$

Hence for all  $a \in A$ ,

$$\text{fold}^{inv} B(a) = \text{inc}_{inv}(\text{fold}^{\mathcal{B}|_{inv}}(a)) = \text{fold}^{\mathcal{B}|_{inv}}(a) \in inv.$$

We conclude that  $inv$  contains  $\text{img}(\text{fold}^{\mathcal{B}})$ .

Alternative proof of (4): By Lemma INV,

$$(\text{fold}^{\mathcal{B}})^{-1}(inv) = \{a \in A \mid \text{fold}^{\mathcal{B}}(a) \in inv\}$$

is a  $\Sigma$ -invariant of  $\mathcal{A}$ . By (3),  $A$  is the only  $\Sigma$ -invariant of  $\mathcal{A}$ .

Hence  $(fold^B)^{-1}(inv) = A$  and thus for all  $b \in inv$  there is  $a \in A$  with  $fold^B(a) = b$ , i.e.,  $b \in img(fold^B)$ . □

## Algebraic induction as fixpoint induction

Let  $C \subseteq F$  be set of constructors,  $C\Sigma = (S, \mathcal{I}, C)$  and for all  $s \in S$ ,  $\varphi_s$  be a  $\Sigma$ -formula such that  $free(\varphi_s) \subseteq \{x\} \subseteq V_s$  and  $\mathcal{C}$  is an initial  $C\Sigma$ -algebra.

Moreover, let  $P = \{inv_e : e \rightarrow 2 \mid e \in \mathcal{T}_p(S, \mathcal{I})\}$ ,  $\Sigma' = (S, \mathcal{I}, F \cup P)$  and  $AX$  be the set of the following Horn clauses for  $P$ :

$$\begin{aligned} inv_s(c(x)) &\Leftarrow inv_e(x), & c : e \rightarrow s \in C, \\ inv_e(\iota_i(x)) &\Leftarrow inv_{e_i}(x), & e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I}), \quad i \in I, \\ inv_e(x) &\Leftarrow \bigwedge_{i \in I} inv_{e_i}(\pi_i(x)), \quad e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I}), \\ inv_B(x), & & B \in \mathcal{I}. \end{aligned}$$

Let  $SP = (\Sigma, P, AX)$  and  $\mathcal{A} \in Alg_{\Sigma', \mathcal{C}}$  be the algebra with  $inv_s^{\mathcal{A}} = \{g(x) \mid g \in \varphi_s^{\mathcal{C}}\}$ . A proof of  $inv_s(x) \Rightarrow \varphi_s$  by fixpoint induction establishes an  $S$ -sorted subset  $B$  of  $inv^{\mathcal{A}} = (inv_s^{\mathcal{A}})_{s \in S}$  such that the algebra  $\mathcal{A}' \leq \mathcal{A}$  with  $inv_s^{\mathcal{A}'} = B_s$  satisfies  $AX$ , i.e.,  $\mathcal{A}' \in Alg_{SP, \mathcal{C}}$ .

By Lemma MUPRED,  $\mu AX =_{\text{def}} \text{lfp}(\Phi_{SP,\mathcal{C}})$  is initial in  $\text{Alg}_{SP,\mathcal{C}}$ . Since  $\mu AX$  interprets  $\text{inv} = (\text{inv}_s)_{s \in S}$  as the least  $C\Sigma$ -invariant of  $\mathcal{C}$ ,  $\text{Alg}_{SP,\mathcal{C}}$  is a subcategory of  $\text{Alg}_{C\Sigma}$  and thus by Lemma INI (4), the least  $C\Sigma$ -invariant of  $\mathcal{C}$  coincides with the image of  $\text{fold}^{\mathcal{A}'} : \mu AX \rightarrow \mathcal{A}'$ ,

$$\text{img}(\text{fold}_s^{\mathcal{A}'}) = \text{inv}_s^{\mu AX} \subseteq \text{inv}_s^{\mathcal{A}'} \subseteq \text{inv}_s^{\mathcal{A}} = \{g(x) \mid g \in \varphi_s^{\mathcal{C}}\}. \quad (1)$$

Moreover, Lemma INI (3) implies that the carrier  $A$  of  $\mathcal{C}$  is the only  $C\Sigma$ -invariant of  $\mathcal{C}$ . Hence by (1),  $A = \text{img}(\text{fold}_s^{\mathcal{A}'}) \subseteq \{g(x) \mid g \in \varphi_s^{\mathcal{C}}\}$ , i.e.,  $\mathcal{C}$  satisfies  $\varphi_s$ .

The number of generalizations of  $\varphi$ , i.e., applications of (2) and consecutive extensions of axioms for the predicate  $q$ , in the proof of  $p(x) \Rightarrow \varphi$  by fixpoint induction agrees with the number of iterations of step 2 in the corresponding proof by algebraic induction. Hence  $q$  is interpreted by the set  $B$  constructed in that proof.

## Invariants are monotone

Suppose that for all  $s \in S$ ,  $F$  contains a **constraint predicate**  $\subset_s : s \rightarrow 2$ .  $\varphi$  is **constraint compatible** if for all  $s \in S$ ,  $x \in V_s$  and subformulas  $\exists x\psi$  or  $\forall x\psi$  of  $\varphi$  there is  $\rho \in \text{Fo}_{\Sigma}(V)$  such that  $\psi = (\subset_s(x) \wedge \rho)$  and  $\psi = (\subset_s(x) \Rightarrow \rho)$ , respectively.

## Lemma INC

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ ,  $inv$  be a  $\Sigma$ -invariant of  $\mathcal{A}$ ,  $\mathcal{B} = \mathcal{A}|_{inv}$  (see chapter 9) and  $\varphi \in Fo_\Sigma(V)$  be constraint compatible such that for all  $s \in S$ ,  $\subset_s^{\mathcal{A}} = inv_s$ .

$$(1) \varphi^{\mathcal{B}} = \{g \in \varphi^{\mathcal{A}} \mid g(V) \subseteq inv\}.$$

$$(2) \mathcal{A} \models \varphi \text{ implies } \mathcal{B} \models \varphi.$$

*Proof of (1) by induction on the size of  $\varphi$ .*

W.l.o.g. the only logical operators of  $\varphi$  are  $\neg$ ,  $\wedge$  and  $\forall$ .

Let  $p : e \rightarrow 2 \in P$  and  $t \in T_\Sigma(V)_e$ .

$$\begin{aligned} p(t)^{\mathcal{B}} &= \{g \in inv^V \mid p^{\mathcal{B}}(g^*(t)) = 1\} = \{g \in A^V \mid g(V) \subseteq inv, p^{\mathcal{A}}(g^*(t)) = 1\} \\ &= \{g \in p(t)^{\mathcal{A}} \mid g(V) \subseteq inv\}. \end{aligned}$$

Let  $\varphi, \psi \in Fo_\Sigma(V)$ ,  $s \in S$  and  $x \in V_s$ .

$$\begin{aligned} (\neg\varphi)^{\mathcal{B}} &= inv^V \setminus \varphi^{\mathcal{B}} \stackrel{ind.\ hyp.}{=} inv^V \setminus \{g \in \varphi^{\mathcal{A}} \mid g(V) \subseteq inv\} = inv^V \setminus \varphi^{\mathcal{A}} \\ &= \{g \in A^V \setminus \varphi^{\mathcal{A}} \mid g(V) \subseteq inv\} = \{g \in (\neg\varphi)^{\mathcal{A}} \mid g(V) \subseteq inv\}, \end{aligned}$$

$$\begin{aligned} (\varphi \wedge \psi)^{\mathcal{B}} &= \varphi^{\mathcal{B}} \cap \psi^{\mathcal{B}} \stackrel{ind.\ hyp.}{=} \{g \in \varphi^{\mathcal{A}} \mid g(V) \subseteq inv\} \cap \{g \in \psi^{\mathcal{A}} \mid g(V) \subseteq inv\} \\ &= \{g \in \varphi^{\mathcal{A}} \cap \psi^{\mathcal{A}} \mid g(V) \subseteq inv\} = \{g \in (\varphi \wedge \psi)^{\mathcal{A}} \mid g(V) \subseteq inv\}. \end{aligned}$$

Let  $\varphi = (\subset_s(x) \Rightarrow \psi)$  for some  $\psi \in \text{Fo}_\Sigma(V)$ . If  $s \in S$ , the form of  $\varphi$  follows by the assumption of constraint compatibility. If  $s \subseteq \mathcal{I}$ , we may also assume that  $\varphi$  has this form because

$$\subset_s(x)^\mathcal{A} = \{g \in A^V \mid g(x) \in \subset_s^\mathcal{A}\} = \{g \in A^V \mid g(x) \in A_s\} = A^V = \text{True}^\mathcal{A}$$

and thus  $\varphi^\mathcal{A} = (\subset_s(x) \Rightarrow \varphi)^\mathcal{A}$ .

Moreover, for all  $g \in A^V$  and  $a \in A_s$ ,

$$\begin{aligned} g[a/x] \in \varphi^\mathcal{A} &\Leftrightarrow g[a/x] \in (\subset_s(x) \Rightarrow \psi)^\mathcal{A} \\ &\Leftrightarrow (g[a/x] \in \subset_s(x)^\mathcal{A} \Rightarrow g[a/x] \in \psi^\mathcal{A}), \end{aligned} \quad (3)$$

$$g[a/x] \in \subset_s(x)^\mathcal{A} \Leftrightarrow g[a/x](x) \in \subset_s^\mathcal{A} \Leftrightarrow a \in \subset_s^\mathcal{A} \Leftrightarrow a \in \text{inv}_s. \quad (4)$$

Hence

$$\begin{aligned} (\forall x \varphi)^\mathcal{B} &= \{g \in \text{inv}^V \mid \forall a \in \text{inv}_s : g[a/x] \in \varphi^\mathcal{B}\} \\ &\stackrel{\text{ind. hyp.}}{=} \{g \in \text{inv}^V \mid g[a/x](V) \subseteq \text{inv}, \forall a \in \text{inv}_s : g[a/x] \in \varphi^\mathcal{A}\} \\ &= \{g \in \text{inv}^V \mid g(V) \subseteq \text{inv}, \forall a \in \text{inv}_s : g[a/x] \in \varphi^\mathcal{A}\} \\ &= \{g \in A^V \mid g(V) \subseteq \text{inv}, \forall a \in \text{inv}_s : g[a/x] \in \varphi^\mathcal{A}\} \\ &= \{g \in A^V \mid g(V) \subseteq \text{inv}, \forall a \in A_s : a \in B_s \Rightarrow g[a/x] \in \varphi^\mathcal{A}\} \end{aligned}$$

$$\begin{aligned}
 &\stackrel{(3)}{=} \{g \in A^V \mid g(V) \subseteq B, \\
 &\quad \forall a \in A_s : a \in B_s \Rightarrow (\textcolor{blue}{g}[a/x] \in \subset_s(x)^{\mathcal{A}} \Rightarrow g[a/x] \in \psi^{\mathcal{A}})\} \\
 &\stackrel{(4)}{=} \{g \in A^V \mid g(V) \subseteq B, \forall a \in A_s : a \in B_s \Rightarrow (\textcolor{blue}{a} \in B_s \Rightarrow g[a/x] \in \psi^{\mathcal{A}})\} \\
 &= \{g \in A^V \mid g(V) \subseteq B, \forall a \in A_s : \textcolor{blue}{a} \in B_s \Rightarrow g[a/x] \in \psi^{\mathcal{A}}\} \\
 &\stackrel{(4)}{=} \{g \in A^V \mid g(V) \subseteq B, \forall a \in A_s : \textcolor{blue}{g}[a/x] \in \subset_s(x)^{\mathcal{A}} \Rightarrow g[a/x] \in \psi^{\mathcal{A}}\} \\
 &\stackrel{(3)}{=} \{g \in A^V \mid g(V) \subseteq B, \forall a \in A_s : g[a/x] \in \varphi^{\mathcal{A}}\} \\
 &= \{g \in (\forall x \varphi)^{\mathcal{A}} \mid g(V) \subseteq B\}.
 \end{aligned}$$

*Proof of (2).*

Suppose that  $\mathcal{A}$  satisfies  $\varphi$ . Then  $\varphi^{\mathcal{A}} = A^V$  and thus by (1),

$$\varphi^{\mathcal{B}} = \{g \in \varphi^{\mathcal{A}} \mid g(V) \subseteq \text{inv}\} = \{g \in \text{inv}^V \mid g \in \varphi^{\mathcal{A}}\} = \{g \in \text{inv}^V \mid g \in A^V\} = \text{inv}^V,$$

i.e.,  $\mathcal{B}$  satisfies  $\varphi$ . □

## Congruences and algebraic coinduction

Let  $\Sigma = (S, \mathcal{I}, F)$  be a signature,  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $R$  be a  $\Sigma$ -congruence on  $\mathcal{A}$ .

$nat_R = (nat_{R_s} : A_s \rightarrow A_s/R_s)_{s \in S}$  denotes the  $S$ -sorted **natural mapping** that sends every element of  $A$  to its equivalence class by  $R$ .

$nat_R$  is an epimorphism in  $Alg_\Sigma$  from  $\mathcal{A}$  to  $\mathcal{A}/R$  (see chapter 9).

For all  $f : e \rightarrow e' \in Arr_\Sigma$  and  $(a, b) \in R_e$ ,

$$nat_R(f^\mathcal{A}(a)) = f^{\mathcal{A}/R}(nat_R(a)) = f^{\mathcal{A}/R}(nat_R(b)) = nat_R(f^\mathcal{A}(b))$$

and thus  $(f^\mathcal{A}(a), f^\mathcal{A}(b)) \in R_{e'}$ . □

### Lemma KER

- (1) Let  $h : A \rightarrow B$  be an  $S$ -sorted function and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ .  $B$  can be extended to a  $\Sigma$ -algebra  $\mathcal{B}$  and  $h$  to a  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $\mathcal{B}$  iff  $ker(h)$  is a  $\Sigma$ -congruence.

(2)  $h : \mathcal{A} \rightarrow \mathcal{B}$  is  $\Sigma$ -homomorphic iff there is a unique  $\Sigma$ -monomorphism  $h' : \mathcal{A}/\ker(h) \rightarrow \mathcal{B}$  with  $h' \circ \text{nat}_{\ker(h)} = h$ .

Hence, if  $h$  is epi, then by Lemma EPIMON (1),  $h'$  is epi and thus  $\mathcal{A}/\ker(h)$  and  $\mathcal{B}$  are  $\Sigma$ -isomorphic.

*Proof.*

(1) If  $h$  is  $\Sigma$ -homomorphic, then  $\ker(h)$  is a  $\Sigma$ -congruence. Let  $\ker(h)$  be a  $\Sigma$ -congruence. For all  $f : e \rightarrow e' \in F$ , define  $f^{\mathcal{B}} : B_e \rightarrow B_{e'}$  such that for all  $a \in A_e$ ,  $f^{\mathcal{B}}(h(a)) = h(f^{\mathcal{A}}(a))$  and for all  $p : e \in P$ , define  $p^{\mathcal{B}} = h(p^{\mathcal{A}})$ . Then  $\mathcal{B}$  is a  $\Sigma$ -algebra and  $h$  is  $\Sigma$ -homomorphic.

(2)  $h'$  with  $h'([a]_{\ker(h)}) = h(a)$  for all  $a \in A$  has all desired properties. The uniqueness and the homomorphism property of  $h'$  follow from Lemma EMH (1).  $\square$

Moreover, by Kleene's Fixpoint Theorem (2), for every  $S$ -sorted binary relation  $R$  on  $A$ , the greatest  $\Sigma$ -congruence contained in  $R$  is the intersection of all  $R_n$ ,  $n \in \mathbb{N}$ , with  $R_0 = R$  and for all  $s \in S$ ,

$$R_{n+1,s} = \{(a, b) \in A^2 \mid \forall d : s \rightarrow e \in D : (d^{\mathcal{A}}(a), d^{\mathcal{A}}(b)) \in R_{n,e}^{eq}\}.$$

## Proofs by algebraic coinduction

- Let  $D \subseteq F$  be a set of destructors and  $D\Sigma = (S, \mathcal{I}, D)$ . A  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  **satisfies the (algebraic) coinduction principle for  $D$**  if for all  $S$ -sorted binary relations  $R$  on  $\mathcal{A}$ ,  $R \subseteq \Delta_A$  iff there is a  $D\Sigma$ -congruence on  $\mathcal{A}$  that contains  $R$ .

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra  $\mathcal{A}$  with carrier  $A$  that satisfies the coinduction principle for  $D$  and for all  $s \in S$ ,  $E_s \subseteq T_\Sigma(V)_s^2$ .

Then the (in)validity of  $E_s$ ,  $s \in S$ , in  $\mathcal{A}$  may be proved by the following iterative algorithm:

- **Step 1:** For all  $s \in S$ , set  $R_s := R_{0,s} =_{\text{def}} \{(g^*(t), g^*(u)) \mid (t, u) \in E_s, g \in A^V\}$ .
- **Step 2:** For all  $s \in S$ , let

$$R'_s = \{(a, b) \in A_s^2 \mid \forall d : s \rightarrow e \in D : (d^\mathcal{A}(a), d^\mathcal{A}(b)) \in R_e^{eq}\}.$$

- **Step 3:** If  $R \subseteq R'$ , then stop: Since  $R'$  is an equivalence relation,  $R \subseteq R'$  implies  $R^{eq} \subseteq R'$ . Consequently,  $R^{eq}$  is a  $D\Sigma$ -congruence that contains  $R_0$ , and thus by the coinduction principle for  $D$ ,  $R_0 \subseteq R^{eq} \subseteq \Delta_A$ . Hence for all  $s \in S$ ,

$$\{(g^*(t), g^*(u)) \mid (t, u) \in E_s, g \in A^V\} = R_{0,s} \subseteq \Delta_{A_s},$$

i.e.,  $\mathcal{A}$  satisfies  $E$ .

If  $R \not\subseteq R'$ , then for all  $e \in \mathcal{T}_p(S, \mathcal{I})$ , set

$$R_e := R_e \cup \{(d^{\mathcal{A}}(a), d^{\mathcal{A}}(b)) \mid (a, b) \in R_s, d : s \rightarrow e \in D\}$$

and go to Step 2.

## Lemma CONG

Let  $h : \mathcal{A} \rightarrow \mathcal{B}$  be  $\Sigma$ -homomorphic and  $R$  be a  $\Sigma$ -congruence on  $\mathcal{A}$ .

$$h(R) = \{(h(a), h(b)) \mid (a, b) \in R\}$$

is a  $\Sigma$ -congruence on  $\mathcal{B}$ .

*Proof.* Let  $f : e \rightarrow e' \in F$  and  $(c, d) \in h(R)_e$ . Then  $c = h(a)$  and  $d = h(b)$  for some  $(a, b) \in R_e$ . Hence  $(f^{\mathcal{A}}(a), f^{\mathcal{A}}(b)) \in R_{e'}$ .

Since  $h$  is  $\Sigma$ -homomorphic,  $f^{\mathcal{B}}(c) = f^{\mathcal{B}}(h(a)) = h(f^{\mathcal{A}}(a))$  and  $f^{\mathcal{B}}(d) = f^{\mathcal{B}}(h(b)) = h(f^{\mathcal{A}}(b))$ . Hence  $(f^{\mathcal{B}}(c), f^{\mathcal{B}}(d)) \in h(R)_{e'}$ .  $\square$

## Lemma FIN (Final algebras satisfy the coninduction principle for their destructors)

Let  $\Sigma = (S, \mathcal{I}, F)$  be a **destructive** signature and  $\mathcal{A}, \mathcal{B}$  be  $\Sigma$ -algebras with carriers  $A, B$  and  $\mathcal{K}$  be a full subcategory  $\mathcal{K}$  of  $Alg_{\Sigma}$  that is closed under quotients.

- (1)  $\mathcal{A}$  satisfies the coinduction principle iff  $\Delta_A$  is the only  $\Sigma$ -congruence on  $\mathcal{A}$ .
- (2) If  $\Delta_A$  is the only  $\Sigma$ -congruence on  $\mathcal{A}$ , then all  $\Sigma$ -homomorphisms from  $\mathcal{B}$  to  $\mathcal{A}$  coincide.
- (3)  $\mathcal{A}$  is final in  $\mathcal{K}$  iff  $\Delta_A$  is the only  $\Sigma$ -congruence on  $\mathcal{A}$  and for all  $\mathcal{B} \in \mathcal{K}$  there is a  $\Sigma$ -homomorphism from  $\mathcal{B}$  to  $\mathcal{A}$ .
- (4) If  $\mathcal{A}$  is final in  $\mathcal{K}$ , then for all  $\mathcal{B} \in \mathcal{K}$ , the kernel of the unique  $\Sigma$ -homomorphism  $\text{unfold}^{\mathcal{B}} : \mathcal{B} \rightarrow \mathcal{A}$  is the greatest  $\Sigma$ -congruence on  $\mathcal{B}$  ([146], Prop. 2.7).

*Proof.*

- (1) “ $\Rightarrow$ ”: Suppose that  $\mathcal{A}$  satisfies the coinduction principle and  $R$  is a  $\Sigma$ -congruence on  $\mathcal{A}$ . Hence  $R \subseteq \Delta_A$ . Since  $R$  is reflexive and thus  $\Delta_A \subseteq R$ ,  $R$  agrees with  $\Delta_A$ .  
 “ $\Leftarrow$ ”: Suppose that  $R$  is a  $\Sigma$ -congruence that contains a binary relation  $R'$  on  $A$  and  $\Delta_A$  is the only  $\Sigma$ -congruence on  $\mathcal{A}$ . Then  $R' \subseteq R = \Delta_A$ .
- (2) Let  $g, h : \mathcal{B} \rightarrow \mathcal{A}$  be  $\Sigma$ -homomorphisms. Then  $R = \{(g(b), h(b)) \mid b \in B\}$  is a  $\Sigma$ -congruence on  $\mathcal{A}$ : Let  $f : e \rightarrow e' \in F$ ,  $b \in B_e$  and  $(g_e(b), h_e(b)) \in R_e$ . Since  $g$  and  $h$  are  $\Sigma$ -homomorphic,  $f^{\mathcal{A}}(g_e(b)) = g_{e'}(f^{\mathcal{B}}(b))$  and  $f^{\mathcal{A}}(h_e(b)) = h_{e'}(f^{\mathcal{B}}(b))$ .

Since  $g$  and  $h$  are  $S$ -sorted, Lemma LIFT (4) implies

$$(f^{\mathcal{A}}(g_e(b)), f^{\mathcal{A}}(h_e(b))) = (g_{e'}(f^{\mathcal{B}}(b)), h_{e'}(f^{\mathcal{B}}(b))) \in R_{e'}.$$

Since  $\Delta_A$  is the only  $\Sigma$ -congruence on  $\mathcal{A}$ ,  $R$  agrees with  $\Delta_A$  and thus for all  $b \in B$ ,  $g(b) = h(b)$ .

(3) “ $\Rightarrow$ ”: Let  $\mathcal{A}$  be final in  $\mathcal{K}$  and  $R$  be a  $\Sigma$ -congruence on  $\mathcal{A}$ .  $R$  induces the  $\Sigma$ -epimorphism  $nat_R : \mathcal{A} \rightarrow \mathcal{A}/R$ . Hence Lemma MINMAX (2) implies that  $nat_R$  is iso in  $\mathcal{K}$  and thus  $R = \Delta_A$ . Since  $\mathcal{A}$  is final in  $\mathcal{K}$ , there is a  $\Sigma$ -homomorphism from  $\mathcal{B}$  to  $\mathcal{A}$ .

“ $\Leftarrow$ ”: Suppose that  $\Delta_A$  is the only  $\Sigma$ -congruence on  $\mathcal{A}$  and for all  $\mathcal{B} \in \mathcal{K}$  there is a  $\Sigma$ -homomorphism  $h : \mathcal{B} \rightarrow \mathcal{A}$ . By (2),  $h$  is unique. Hence  $\mathcal{A}$  is final in  $\mathcal{K}$ .

(4) Let  $R$  be a  $\Sigma$ -congruence on  $\mathcal{B}$ . Since  $\mathcal{A}$  is final in  $\mathcal{K}$ , the following diagram commutes:

$$\begin{array}{ccc} \mathcal{B} & \xrightarrow{\text{unfold}^{\mathcal{B}}} & \mathcal{A} \\ & \searrow nat_R & \swarrow \text{unfold}^{\mathcal{B}/R} \\ & \mathcal{B}/R & \end{array}$$

Hence for all  $b, c \in B$ ,

$$\begin{aligned} (b, c) \in R &\Rightarrow [b]_R = [c]_R \Rightarrow \text{unfold}^{\mathcal{B}}(b) = \text{unfold}^{\mathcal{B}/R}([b]_R) = \text{unfold}^{\mathcal{B}/R}([c]_R) \\ &= \text{unfold}^{\mathcal{B}}(c). \end{aligned}$$

We conclude that  $\ker(\text{unfold}^{\mathcal{B}})$  contains  $R$ .

Alternative proof of (4): By Lemma CONG,

$$\text{unfold}^{\mathcal{B}}(R) = \{(\text{unfold}^{\mathcal{B}}(b), \text{unfold}^{\mathcal{B}}(c)) \mid (b, c) \in R\}$$

is a  $\Sigma$ -congruence on  $\mathcal{A}$ . By (3),  $\Delta_A$  is the only  $\Sigma$ -congruence on  $\mathcal{A}$ . Hence  $\text{unfold}^{\mathcal{B}}(R) = \Delta_A$  and thus for all  $(b, c) \in R$ ,  $\text{unfold}^{\mathcal{B}}(b) = \text{unfold}^{\mathcal{B}}(c)$ , i.e.,  $(b, c) \in \ker(\text{unfold}^{\mathcal{B}})$ .  $\square$

## Algebraic coinduction as fixpoint coinduction

Let  $D \subseteq F$  be a set of destructors,  $D\Sigma = (S, \mathcal{I}, D)$  and for all  $s \in S$ ,  $E_s \subseteq T_{\Sigma}(V)_s^2$ , such that  $\mathcal{C}$  is a final  $D\Sigma$ -algebra.

Moreover, let  $P = \{\sim_e : e \times e \mid e \in \mathcal{T}_p(S, \mathcal{I})\}$ ,  $\Sigma' = (S, F \cup P)$  and  $AX$  is the following set of co-Horn clauses for  $P$ :

$$x \sim_s y \Rightarrow d(x) \sim_e d(y), \quad d : s \rightarrow e \in D,$$

$$\begin{aligned}
 \iota_i(x) \sim_e \iota_i(y) &\Rightarrow x \sim_{e_i} y, \quad e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I}), \quad i \in I, \\
 \iota_i(x) \sim_e \iota_j(y) &\Rightarrow \text{False}, \quad e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I}), \quad i, j \in I, \quad i \neq j, \\
 x \sim_e y &\Rightarrow \pi_i(x) \sim_{e_i} \pi_i(y), \quad e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I}), \quad i \in I, \\
 x \sim_B y &\Rightarrow x = y, \quad B \in \mathcal{I}.
 \end{aligned}$$

Let  $SP = (\Sigma, P, AX)$ ,  $A$  be the carrier of  $\mathcal{C}$  and  $\mathcal{A} \in Alg_{\Sigma', \mathcal{C}}$  be the algebra with

$$\sim_s^{\mathcal{A}} = \{(g^*(t), g^*(u)) \mid t = u \in E_s, \quad g \in A^V\}$$

and for all  $s \in S$ , let  $x, y \in V_s \setminus var(E_s)$  and

$$\varphi_s = \bigvee_{t=u \in E_s} \exists var(E_s) : (x = t \wedge y = u).$$

A proof of  $E_s$  by fixpoint coinduction starts out from  $\varphi_s \Rightarrow x \sim_s y$  and establishes an  $S$ -sorted binary relation  $R$  that contains  $\sim^{\mathcal{A}} = (\sim_s^{\mathcal{A}})_{s \in S}$  such that the algebra  $\mathcal{A}' \geq \mathcal{A}$  with  $\sim_s^{\mathcal{A}'} = R_s$  satisfies  $AX$ , i.e.,  $\mathcal{A}' \in Alg_{SP, \mathcal{C}}$ .

By Lemma NUPRED,  $\nu AX =_{def} gfp(\Phi_{SP, \mathcal{C}})$  is final in  $Alg_{SP, \mathcal{C}}$ . Since  $\nu AX$  interprets  $\sim = (\sim_s)_{s \in S}$  as the greatest  $D\Sigma$ -congruence on  $\mathcal{C}$ ,  $Alg_{SP, \mathcal{C}}$  is a subcategory of  $Alg_{D\Sigma}$  and thus by Lemma FIN (4), the greatest  $D\Sigma$ -congruence on  $\mathcal{C}$  coincides with the kernel of  $\text{unfold}^{\mathcal{A}'} : \mathcal{A}' \rightarrow \nu AX$ ,

$$\{(g^*(t), g^*(u)) \mid t = u \in E_s, g \in A^V\} = \sim_s^{\mathcal{A}} \subseteq \sim_s^{\mathcal{A}'} \subseteq \sim_s^{\nu AX} = \ker(\text{unfold}^{\mathcal{A}'}) \quad (1)$$

Moreover, Lemma FIN (3) implies that  $\Delta_A^2$  is the only  $D\Sigma$ -congruence on  $\mathcal{C}$ . Hence by (1),  $\{(g^*(t), g^*(u)) \mid t = u \in E_s, g \in A^V\} \subseteq \ker(\text{unfold}^{\mathcal{A}'}) = \Delta_A^2$ , i.e.,  $\mathcal{C}$  satisfies  $E_s$ .

The number of generalizations of  $\varphi$ , i.e., applications of (2) and consecutive extensions of axioms for the predicate  $q$ , in the proof of  $\varphi \Rightarrow p(x, y)$  by fixpoint coinduction agrees with the number of iterations of step 2 in the corresponding proof by algebraic coinduction. Hence  $q$  is interpreted by the relation  $R$  constructed in that proof.

**Example ZIP** Let  $\Sigma = \text{Stream}(\mathbb{Z}) \cup F$  where

$$\begin{aligned} F = & \{ \text{zeros}, \text{ones}, \text{blink} : 1 \rightarrow \text{list} \} \cup \\ & \{ \text{cons} : \mathbb{N} \times \text{list} \rightarrow \text{list}, \text{zip} : \text{list} \times \text{list} \rightarrow \text{list}, \text{evens} : \text{list} \rightarrow \text{list} \}. \end{aligned}$$

Let  $\mathcal{C}$  be a  $\Sigma$ -algebra such that  $\mathcal{C}|_{\text{Stream}(\mathbb{Z})}$  is final in  $\text{Alg}_{\text{Stream}(\mathbb{Z})}$  and satisfies the following equations:

$$\begin{aligned} \text{head}(\text{zeros}) &= 0, & \text{tail}(\text{zeros}) &= \text{zeros}, \\ \text{head}(\text{ones}) &= 1, & \text{tail}(\text{ones}) &= \text{ones}, \\ \text{head}(\text{blink}) &= 0, & \text{tail}(\text{blink}) &= \text{cons}(1, \text{blink}), \\ \text{head}(\text{cons}(x, s)) &= x, & \text{tail}(\text{cons}(x, s)) &= s, \end{aligned}$$

$$\begin{aligned} \text{head}(\text{zip}(s, s')) &= \text{head}(s'), \quad \text{tail}(\text{zip}(s, s')) = \text{zip}(s', s), \\ \text{head}(\text{evens}(s)) &= \text{head}(s), \quad \text{tail}(\text{evens}(s)) = \text{tail}(\text{tail}(s)). \end{aligned}$$

We show by algebraic coinduction that  $\mathcal{A}$  satisfies the equations

$$\text{zip}(\text{zeros}, \text{ones}) = \text{blink}, \tag{1}$$

$$\text{evens}(\text{zip}(s, s')) = s. \tag{2}$$

Let

$$R_0 = \{(g^*(t), g^*(t')) \mid t = t' \in \{(1), (2)\}, g \in A^V\},$$

$$R'_0 = \{(a, b) \in A^2 \mid \text{head}^{\mathcal{A}}(a) = \text{head}^{\mathcal{A}}(b), (\text{tail}^{\mathcal{A}}(a), \text{tail}^{\mathcal{A}}(b)) \in R_0^{eq}\}$$

and  $(a, b) \in R'_0$ . Then there are  $t = t' \in \{(1), (2)\}$  and  $g \in A^V$  such that  $a = g^*(t)$  and  $b = g^*(t')$ .

*Case 1:*  $t = \text{zip}(\text{zeros}, \text{ones})$  and  $t' = \text{blink}$ . Then

$$\begin{aligned} \text{head}^{\mathcal{A}}(a) &= \text{head}^{\mathcal{A}}(g^*(\text{zip}(\text{zeros}, \text{ones}))) = \text{head}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(\text{zeros}^{\mathcal{A}}, \text{ones}^{\mathcal{A}})) \\ &= \text{head}^{\mathcal{A}}(\text{blink}^{\mathcal{A}}) = \text{head}^{\mathcal{A}}(g^*(\text{blink})) = \text{head}^{\mathcal{A}}(b), \end{aligned} \tag{3}$$

$$\begin{aligned} \text{tail}^{\mathcal{A}}(a) &= \text{tail}^{\mathcal{A}}(g^*(\text{zip}(\text{zeros}, \text{ones}))) = \text{tail}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(\text{zeros}^{\mathcal{A}}, \text{ones}^{\mathcal{A}}))) \\ &= \text{zip}^{\mathcal{A}}(\text{ones}^{\mathcal{A}}, \text{zeros}^{\mathcal{A}}), \end{aligned} \quad (4)$$

$$\text{tail}^{\mathcal{A}}(b) = \text{tail}^{\mathcal{A}}(g^*(\text{blink})) = \text{tail}^{\mathcal{A}}(\text{blink}^{\mathcal{A}}) = \text{cons}^{\mathcal{A}}(1, \text{blink}^{\mathcal{A}}). \quad (5)$$

Case 2:  $t = \text{evens}(\text{zip}(s, s'))$  and  $t' = s$ . Then

$$\begin{aligned} \text{head}^{\mathcal{A}}(a) &= \text{head}^{\mathcal{A}}(g^*(\text{evens}(\text{zip}(s, s')))) = \text{head}^{\mathcal{A}}(\text{evens}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(g(s), g(s')))) \\ &= \text{head}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(g(s), g(s'))) = \text{head}^{\mathcal{A}}(g(s)) = \text{head}^{\mathcal{A}}(g^*(s)) = \text{head}^{\mathcal{A}}(b), \end{aligned} \quad (6)$$

$$\begin{aligned} \text{tail}^{\mathcal{A}}(a) &= \text{tail}^{\mathcal{A}}(g^*(\text{evens}(\text{zip}(s, s')))) = \text{tail}^{\mathcal{A}}(\text{evens}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(g(s), g(s')))) \\ &= \text{evens}^{\mathcal{A}}(\text{tail}^{\mathcal{A}}(\text{tail}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(g(s), g(s'))))) = \text{evens}^{\mathcal{A}}(\text{tail}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(g(s'), \text{tail}(g(s))))) \\ &= \text{evens}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(\text{tail}^{\mathcal{A}}(g(s)), \text{tail}^{\mathcal{A}}(g(s')))) = \text{evens}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(\text{tail}^{\mathcal{A}}(g^*(s)), \text{tail}^{\mathcal{A}}(g(s')))) \\ &= \text{evens}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(\text{tail}^{\mathcal{A}}(b), \text{tail}^{\mathcal{A}}(g(s')))). \end{aligned} \quad (7)$$

Let  $h(s) = \text{tail}^{\mathcal{A}}(b)$  and  $h(s') = \text{tail}^{\mathcal{A}}(g(s'))$ . Then by (7),

$$\begin{aligned} (\text{tail}^{\mathcal{A}}(a), \text{tail}^{\mathcal{A}}(b)) &= (\text{evens}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(\text{tail}^{\mathcal{A}}(b), \text{tail}^{\mathcal{A}}(g(s')))), \text{tail}^{\mathcal{A}}(b)) \\ &= (\text{evens}^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(h(s), h(s'))), h(s)) = (h^*(\text{evens}(\text{zip}(s, s'))), h^*(s)) \in R_0 \end{aligned} \quad (8).$$

However, in Case 1, by (4) and (5),  $(tail^{\mathcal{A}}(a), tail^{\mathcal{A}}(b)) \notin R_0^{eq}$  and thus  $R_0 \not\subseteq R'_0$ . Hence we extend  $R_0$  to  $R_1$  as follows—according to Step 3 of the coinductive-proof procedure:

$$a' = \text{zip}^{\mathcal{A}}(\text{ones}^{\mathcal{A}}, \text{zeros}^{\mathcal{A}}), \quad b' = \text{cons}^{\mathcal{A}}(1, \text{blink}^{\mathcal{A}}),$$

$$R_1 = R_0 \cup \{(a', b')\},$$

$$R'_1 = \{(a, b) \in A^2 \mid head^{\mathcal{A}}(a) = head^{\mathcal{A}}(b), (tail^{\mathcal{A}}(a), tail^{\mathcal{A}}(b)) \in R_1^{eq}\}.$$

By (3), (4), (5), (6) and (8),  $R_0 \subseteq R'_1$ . Moreover,

$$\begin{aligned} head^{\mathcal{A}}(a') &= head^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(\text{ones}^{\mathcal{A}}, \text{zeros}^{\mathcal{A}}))) = 1 = head^{\mathcal{A}}(\text{cons}^{\mathcal{A}}(1, \text{blink}^{\mathcal{A}})) \\ &= head^{\mathcal{A}}(b'), \end{aligned}$$

$$\begin{aligned} tail^{\mathcal{A}}(a') &= tail^{\mathcal{A}}(\text{zip}^{\mathcal{A}}(\text{ones}^{\mathcal{A}}, \text{zeros}^{\mathcal{A}})) = \text{zip}^{\mathcal{A}}(\text{zeros}^{\mathcal{A}}, tail^{\mathcal{A}}(\text{ones}^{\mathcal{A}})) \\ &= \text{zip}^{\mathcal{A}}(\text{zeros}^{\mathcal{A}}, \text{ones}^{\mathcal{A}}), \end{aligned}$$

$$tail^{\mathcal{A}}(b') = tail^{\mathcal{A}}(\text{cons}^{\mathcal{A}}(1, \text{blink}^{\mathcal{A}})) = \text{blink}^{\mathcal{A}}.$$

Hence  $(a', b') \in R'_1$ . Consequently,  $R_1 = R_0 \cup \{(a', b')\} \subseteq R'_1$  and thus by the coinduction principle for  $\Sigma$ ,  $R_0 \subseteq \Delta_A^2$ , i.e.,  $\mathcal{A}$  satisfies (1) and (2).

Proofs of (1) and (2) by fixpoint coinduction, performed by `Expander2` with respect to the specification `stream`, can be found [here](#). □

## Coinduction modulo constructors

Let  $D \subseteq F$  be a set of destructors,  $D\Sigma = (S, \mathcal{I}, D)$ ,  $C \subseteq F$  be a set of constructors and  $C\Sigma = (S, \mathcal{I}, C)$ . Under the assumptions of Lemma `MOD` (see chapter 13), the coinduction principle for  $D$  (see above) can be weakened as follows:

- ✿ A  $\Sigma$ -algebra  $A$  with carrier  $A$  **satisfies the coinduction principle for  $D$  modulo  $C$**  if for all  $S$ -sorted binary relations  $R$  on  $\mathcal{A}$ ,  $R \subseteq \Delta_A$  iff there is a  $D\Sigma$ -congruence on  $\mathcal{A}$  **modulo  $C$**  that contains  $R$ .

Accordingly, the above coinductive-proof procedure becomes a method for proving equations by **coinduction modulo  $C$**  if Step 2 is adapted as follows:

- **Step 2:** For all  $s \in S$ , let

$$R'_s = \{(a, b) \in A_s^2 \mid \forall d : s \rightarrow e \in D : (d^{\mathcal{A}}(a), d^{\mathcal{A}}(b)) \in R_{C,e}\}.$$

If  $R \subseteq R'$ , then  $R$  is a  $D\Sigma$ -bisimulation modulo  $C$ . Hence by Lemma `MOD`,  $R_C$  is a  $D\Sigma$ -congruence.

Since  $R_C$  contains  $R_0$ , the coinduction principle for  $D$  implies  $R_0 \subseteq R_C \subseteq \Delta_A$ . Hence for all  $s \in S$ ,

$$\{(g^*(t), g^*(u)) \mid (t, u) \in E_s, g \in A^V\} = R_{0,s} \subseteq \Delta_{A_s},$$

i.e.,  $\mathcal{A}$  satisfies  $E$ .

## Example SEQPAR

Let  $\mathcal{A}$  be the  $Reg(X) \cup Acc(X)$ -algebra with  $\mathcal{A}|_{Reg(X)} = Pow(X)$  and  $\mathcal{A}|_{Acc(X)} = Lang(X)$  (see sample algebras 19 and 20; the sorts *state* and *reg* are identified with each other).

The biinductive definition of the Brzozowski automaton (see Sample biinductively defined functions) provide the assumptions of Lemma MOD. We show that  $\mathcal{A}$  satisfies the distributive law

$$seq(x, par(y, z)) = par(seq(x, y), seq(x, z)). \quad (1)$$

The following proof by coinduction modulo  $\{par\}$  uses the equations that define the Brzozowski automaton (see sample algebra 23) and (2)-(6):

$$par(x, \bar{\emptyset}) = x \tag{2}$$

$$par(par(x_1, y_1), par(x_2, y_2)) = par(par(x_1, x_2), par(y_1, y_2)) \tag{3}$$

$$par(x, ite(b, y, z)) = ite(b, par(x, y), par(x, z)) \tag{4}$$

$$par(ite(b, x, y), z) = ite(b, par(x, z), par(y, z)) \tag{5}$$

$$x * (y + z) = x * y + x * z \tag{6}$$

Let

$$R = \{(g^*(seq(x, par(y, z))), g^*(par(seq(x, y), seq(x, z))) \mid g \in A^V\},$$

$$R' = \{(a, b) \in A^2 \mid \beta^A(a) = \beta^A(b), (\delta^A(a), \delta^A(b)) \in R_C\}$$

and  $(a, b) \in R$ . Then there is  $g \in A^V$  such that  $a = g^*(seq(x, par(y, z)))$  and  $b = g^*(par(seq(x, y), seq(x, z)))$ . Let

$$a_1 = seq^A(\delta^A(g(x))(x'), par^A(g(y), g(z))),$$

$$a_2 = par^A(seq^A(\delta^A(g(x))(x'), g(y)), seq^A(\delta^A(g(x))(x'), g(z))),$$

$$a_3 = par^A(\delta^A(g(y))(x'), \delta^A(g(z))(x')).$$

Hence for all  $x' \in X$ ,

$$\begin{aligned}
 \delta^{\mathcal{A}}(a)(x') &= \delta^{\mathcal{A}}(g^*(seq(x, par(y, z))))(x') = \delta^{\mathcal{A}}(seq^{\mathcal{A}}(g(x), par^{\mathcal{A}}(g(y), g(z))))(x') \\
 &= par^{\mathcal{A}}(seq^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), par^{\mathcal{A}}(g(y), g(z)))), \\
 &\quad ite^{\mathcal{A}}(\beta^{\mathcal{A}}(g(x)), \delta^{\mathcal{A}}(par^{\mathcal{A}}(g(y), g(z)))(x), \emptyset)) \\
 &= par^{\mathcal{A}}(a_1, ite^{\mathcal{A}}(\beta^{\mathcal{A}}(g(x)), \delta^{\mathcal{A}}(par^{\mathcal{A}}(g(y), g(z)))(x), \emptyset)) \\
 &= par^{\mathcal{A}}(a_1, ite^{\mathcal{A}}(\beta^{\mathcal{A}}(g(x)), par^{\mathcal{A}}(\delta^{\mathcal{A}}(g(y))(x'), \delta^{\mathcal{A}}(g(z))(x')), \emptyset)) \\
 &= par^{\mathcal{A}}(a_1, ite^{\mathcal{A}}(\beta^{\mathcal{A}}(g(x)), par^{\mathcal{A}}(\delta^{\mathcal{A}}(g(y))(x'), \delta^{\mathcal{A}}(g(z))(x')), \emptyset)) \\
 &\stackrel{(2),(4)}{=} \begin{cases} par^{\mathcal{A}}(a_1, par^{\mathcal{A}}(\delta^{\mathcal{A}}(g(y))(x'), \delta^{\mathcal{A}}(g(z))(x'))) & \text{if } \beta^{\mathcal{A}}(g(x)) = 1 \\ a_1 & \text{otherwise} \end{cases} \\
 &= \begin{cases} par^{\mathcal{A}}(a_1, a_3) & \text{if } \beta^{\mathcal{A}}(g(x)) = 1 \\ a_1 & \text{otherwise,} \end{cases}
 \end{aligned}$$

$$\begin{aligned}
 \delta^{\mathcal{A}}(b)(x') &= \delta^{\mathcal{A}}(g^*(par(seq(x, y), seq(x, z))))(x') \\
 &= \delta^{\mathcal{A}}(par^{\mathcal{A}}(seq^{\mathcal{A}}(g(x), g(y)), seq^{\mathcal{A}}(g(x), g(z))))(x') \\
 &= par^{\mathcal{A}}(\delta^{\mathcal{A}}(seq^{\mathcal{A}}(g(x), g(y)))(x'), \delta^{\mathcal{A}}(seq^{\mathcal{A}}(g(x), g(z)))(x'))
 \end{aligned}$$

$$\begin{aligned}
&= \text{par}^{\mathcal{A}}(\text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(y)), \text{ite}(\beta^{\mathcal{A}}(g(x)), \delta^{\mathcal{A}}(g(y))(x'), \emptyset)), \\
&\quad \text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(z)), \text{ite}(\beta^{\mathcal{A}}(g(x)), \delta^{\mathcal{A}}(g(z))(x'), \emptyset))) \\
&\stackrel{(2),(4),(5)}{=} \begin{cases} \text{par}^{\mathcal{A}}(\text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(y)), \delta^{\mathcal{A}}(g(y))(x')), \\ \quad \text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(z)), \delta^{\mathcal{A}}(g(z))(x'))) & \text{if } \beta^{\mathcal{A}}(g(x)) = 1 \\ \text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(y)), \text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(z))) & \text{otherwise} \end{cases} \\
&= \begin{cases} \text{par}^{\mathcal{A}}(\text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(y)), \delta^{\mathcal{A}}(g(y))(x')), \\ \quad \text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(z)), \delta^{\mathcal{A}}(g(z))(x')) & \text{if } \beta^{\mathcal{A}}(g(x)) = 1 \\ a_2 & \text{otherwise} \end{cases} \\
&\stackrel{(3)}{=} \begin{cases} \text{par}^{\mathcal{A}}(\text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(y))), \text{seq}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(z)), \\ \quad \text{par}^{\mathcal{A}}(\delta^{\mathcal{A}}(g(y))(x'), \delta^{\mathcal{A}}(g(z))(x')) & \text{if } \beta^{\mathcal{A}}(g(x)) = 1 \\ a_2 & \text{otherwise} \end{cases} \\
&= \begin{cases} \text{par}^{\mathcal{A}}(a_2, a_3) & \text{if } \beta^{\mathcal{A}}(g(x)) = 1 \\ a_2 & \text{otherwise} \end{cases}
\end{aligned}$$

and thus

$$(\delta^{\mathcal{A}}(a)(x'), \delta^{\mathcal{A}}(b)(x')) = \begin{cases} (par^{\mathcal{A}}(a_1, a_3), par^{\mathcal{A}}(a_2, a_3)) & \text{if } \beta^{\mathcal{A}}(g(x)) = 1 \\ (a_1, a_2) & \text{otherwise.} \end{cases} \quad (7)$$

Let  $g' \in A^V$  be defined as follows:  $g'(x) = \delta^{\mathcal{A}}(g(x))(x')$  and for all  $v \in V \setminus \{x\}$ ,  $g'(v) = g(v)$ .

$$\begin{aligned} a_1 &= seq^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), par^{\mathcal{A}}(g(y), g(z))) = seq^{\mathcal{A}}(g'(x), par^{\mathcal{A}}(g'(y), g'(z))) \\ &= g'^*(seq(x, par(y, z))), \\ a_2 &= par^{\mathcal{A}}(seq^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(y)), seq^{\mathcal{A}}(\delta^{\mathcal{A}}(g(x))(x'), g(z))) \\ &= par^{\mathcal{A}}(seq^{\mathcal{A}}(g'(x), g'(y)), seq^{\mathcal{A}}(g'(x), g(z))) = g'^*(par(seq(x, y), seq(x, z))) \end{aligned}$$

and thus  $(a_1, a_2) \in R$ . Hence (7) implies  $(\delta^{\mathcal{A}}(a)(x'), \delta^{\mathcal{A}}(b)(x')) \in R_{\{par\}}$ . Moreover,

$$\begin{aligned} \beta^{\mathcal{A}}(a) &= \beta^{\mathcal{A}}(g^*(seq(x, par(y, z)))) = \beta^{\mathcal{A}}(seq^{\mathcal{A}}(g(x), par^{\mathcal{A}}(g(y), g(z)))) \\ &= \beta^{\mathcal{A}}(g(x)) * \beta^{\mathcal{A}}(par^{\mathcal{A}}(g(y), g(z))) = \beta^{\mathcal{A}}(g(x)) * max(\beta^{\mathcal{A}}(g(y)), \beta^{\mathcal{A}}(g(z))) \\ &\stackrel{(6)}{=} max(\beta^{\mathcal{A}}(g(x)) * \beta^{\mathcal{A}}(g(y)), \beta^{\mathcal{A}}(g(x)) * \beta^{\mathcal{A}}(g(z))) \\ &= max(\beta^{\mathcal{A}}(seq^{\mathcal{A}}(g(x), g(y))), \beta^{\mathcal{A}}(seq^{\mathcal{A}}(g(x), g(z)))) \end{aligned}$$

$$\begin{aligned}
&= \beta^{\mathcal{A}}(\text{par}^{\mathcal{A}}(\text{seq}^{\mathcal{A}}(g(x), g(y)), \text{seq}^{\mathcal{A}}(g(x), g(z)))) \\
&= \beta^{\mathcal{A}}(g^*(\text{par}(\text{seq}(x, y), \text{seq}(x, z)))) = \beta^{\mathcal{A}}(b).
\end{aligned}$$

Hence  $R \subseteq R'$  and thus by the coinduction principle for  $\Sigma$ ,  $R \subseteq \Delta_A^2$ , i.e.,  $\mathcal{A}$  satisfies (1). A proof of (1) by fixpoint coinduction modulo  $\{\text{par}\}$ , performed by [Expander2](#) with respect to the specification [brozowski](#), can be found [here](#).  $\square$

## Quotients are monotone

### Lemma NAT

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ ,  $R$  be a  $\Sigma$ -congruence on  $\mathcal{A}$ ,  $\mathcal{B} = \mathcal{A}/R$  (see chapter 9),  $\varphi \in \text{Fo}_{\Sigma}(V)$  and  $f, g \in A^V$  such that for all  $x \in V$ ,  $(f(x), g(x)) \in R$ .

(1) For all  $t \in T_{\Sigma}(V)_e$ ,  $(f^*(t), g^*(t)) \in R_e$ .

(2)  $f \in \varphi^{\mathcal{A}}$  implies  $g \in \varphi^{\mathcal{A}}$ .

(3)  $\varphi^{\mathcal{B}} = \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}}\}$ .

(4)  $\mathcal{A} \models \varphi$  implies  $\mathcal{B} \models \varphi$ .

*Proof of (1).* Induction on the size of  $t$ .

*Proof of (2) by induction on the size of  $\varphi$ .*

W.l.o.g. the only logical operators of  $\varphi$  are  $\neg$ ,  $\wedge$  and  $\forall$ .

Let  $p : e \rightarrow 2 \in P$ ,  $t \in T_\Sigma(V)_e$  and  $f \in p(t)^\mathcal{A}$ . Then  $f^*(t) \in p^\mathcal{A}$ . By (1),  $(f^*(t), g^*(t)) \in R_e$ . Hence  $g^*(t) \in p^\mathcal{A}$  and thus  $g \in p(t)^\mathcal{A}$ .

Let  $\varphi, \psi \in \text{Fo}_\Sigma(V)$ ,  $s \in S$  and  $x \in V_s$ .

$$f \in (\neg\varphi)^\mathcal{A} \Leftrightarrow f \in A^V \setminus \varphi^\mathcal{A} \stackrel{\text{ind. hyp.}}{\Leftarrow} g \in A_e \setminus \varphi^\mathcal{A} \Leftrightarrow g \in (\neg\varphi)^\mathcal{A},$$

$$f \in (\varphi \wedge \psi)^\mathcal{A} \Leftrightarrow f \in \varphi^\mathcal{A} \cap \psi^\mathcal{A} \stackrel{\text{ind. hyp.}}{\Leftarrow} g \in \varphi^\mathcal{A} \cap \psi^\mathcal{A} \Leftrightarrow g \in (\varphi \wedge \psi)^\mathcal{A},$$

$$f \in (\forall x\varphi)^\mathcal{A} \Leftrightarrow \forall a \in A_s : f[a/x] \in \varphi^\mathcal{A} \stackrel{\text{ind. hyp.}}{\Leftarrow} \forall a \in A_s : g[c/x] \in \varphi^\mathcal{A} \Leftrightarrow g \in (\forall x\varphi)^\mathcal{A}.$$

*Proof of (3) by induction on the size of  $\varphi$ .*

W.l.o.g. the only logical operators of  $\varphi$  are  $\neg$ ,  $\wedge$  and  $\forall$ .

Let  $p : e \rightarrow 2 \in P$  and  $t \in T_\Sigma(V)_e$ .

$$\begin{aligned} p(t)^\mathcal{B} &= \{f \in (A/R)^V \mid f^*(t) \in p^\mathcal{B}\} \\ &= \{nat_R \circ g \mid g \in A^V, (nat_R \circ g)^*(t) \in p^\mathcal{B}\} \\ &= \{nat_R \circ g \mid g \in A^V, nat_R(g^*(t)) \in p^\mathcal{B}\} \end{aligned}$$

$$\begin{aligned}
 &= \{nat_R \circ g \mid g \in A^V, [g^*(t)]_R \in p^{\mathcal{B}}\} \\
 &\stackrel{(1)}{=} \{nat_R \circ g \mid g \in A^V, g^*(t) \in p^{\mathcal{A}}\} \\
 &= \{nat_R \circ g \mid g \in p(t)^{\mathcal{A}}\}.
 \end{aligned}$$

Let  $\varphi, \psi \in Fo_{\Sigma}(V)$ ,  $s \in S$  and  $x \in V_s$ .

$$\begin{aligned}
 (\neg\varphi)^{\mathcal{B}} &= (A/R)^V \setminus \varphi^{\mathcal{B}} \stackrel{ind. \ hyp.}{=} (A/R)^V \setminus \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}}\} \\
 &= \{nat_R \circ g \mid g \in A^V \setminus \varphi^{\mathcal{A}}\} = \{nat_R \circ g \mid g \in (\neg\varphi)^{\mathcal{A}}\}, \\
 (\varphi \wedge \psi)^{\mathcal{B}} &= \varphi^{\mathcal{B}} \cap \psi^{\mathcal{B}} \stackrel{ind. \ hyp.}{=} \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}}\} \cap \{nat_R \circ g \mid g \in \psi^{\mathcal{A}}\} \\
 &= \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}} \cap \psi^{\mathcal{A}}\} = \{nat_R \circ g \mid g \in (\varphi \wedge \psi)^{\mathcal{A}}\}, \\
 (\forall x\varphi)^{\mathcal{B}} &= \{f \in (A/R)^V \mid \forall [a]_R \in (A/R)_s : f[[a]_R/x] \in \varphi^{\mathcal{B}}\} \\
 &= \{nat_R \circ g \mid g \in A^V, \forall [a]_R \in (A/R)_s : (nat_R \circ g)[[a]_R/x] \in \varphi^{\mathcal{B}}\} \\
 &= \{nat_R \circ g \mid g \in A^V, \forall a \in A_s : nat_R \circ g[a/x] \in \varphi^{\mathcal{B}}\} \\
 &\stackrel{ind. \ hyp.}{=} \{nat_R \circ g \mid g \in A^V, \forall a \in A_s : g[a/x] \in \varphi^{\mathcal{A}}\} \\
 &= \{nat_R \circ g \mid g \in (\forall x\varphi)^{\mathcal{A}}\}.
 \end{aligned}$$

*Proof of (4).* Suppose that  $\mathcal{A}$  satisfies  $\varphi$ . Then  $\varphi^{\mathcal{A}} = A^V$  and thus by (3),

$$\varphi^{\mathcal{B}} = \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}}\} = \{nat_R \circ g \mid g \in A^V\} = (A/R)^V,$$

i.e.,  $\mathcal{B}$  satisfies  $\varphi$ .



### Initial models of constructive polynomial signatures

Let  $\Sigma = (S, \mathcal{I}, F)$  be a constructive polynomial signature,  $\kappa$  be the cardinality of the greatest exponent occurring in the source of some  $f \in F$  and  $\lambda$  be the first regular cardinal number  $> \kappa$ .

By Theorem **CONTYPES** (2),  $H_\Sigma$  is  $\lambda$ -cocontinuous and thus by Theorem **INIALG**,  $Alg_{H_\Sigma}$  has an initial object  $\alpha : H_\Sigma(\mu\Sigma) \rightarrow \mu\Sigma$ . In other words,  $\mu\Sigma$  is the initial  $\Sigma$ -algebra (see (1)).

Since  $\mu\Sigma$  is the colimit of the  $\lambda$ -chain  $\mathcal{D}$  of  $Set^S$  defined in Theorem **INIALG**, the **Quotient Theorem** implies that for all  $s \in S$ ,

$$\mu\Sigma_s = (\coprod_{i < \lambda} \mathcal{D}(i)_s) / \sim_s$$

where  $\sim_s$  is the equivalence closure of

$$\{(a, \mathcal{D}(i, i+1)(a)) \mid a \in \mathcal{D}(i)_s, i < \lambda\}.$$

Let  $A$  be a  $\Sigma$ -algebra. The unique  $\Sigma$ -homomorphism  $\text{fold}^{\mathcal{A}} : \mu\Sigma \rightarrow A$  is the unique  $S$ -sorted function such that

$$\coprod_{i < \lambda} \mathcal{D}(i) \xrightarrow{[\beta_i]_{i < \lambda}} A = \coprod_{i < \lambda} \mathcal{D}(i) \xrightarrow{\text{nat} \sim} \mu\Sigma \xrightarrow{\text{fold}^{\mathcal{A}}} A$$

where  $\beta_0$  is the unique  $S$ -sorted function from  $\mathcal{D}(0)$  to  $A$  and for all  $i < \lambda$  and  $s \in S$ ,

$$\beta_{i+1,s} = [f^{\mathcal{A}} \circ F_e(\beta_{i,s})]_{f:e \rightarrow s \in F} : \mathcal{D}(i+1)_s \rightarrow A_s.$$

$H_\Sigma$  is  $\omega$ -cocontinuous and its object mapping reads as follows:

For all  $S$ -sorted sets  $A$  and  $s \in S$ ,

$$\begin{aligned} H_\Sigma(A)_s &= \coprod_{f:e_1 \times \dots \times e_n \rightarrow s \in F} \prod_{i=1}^n A_{e_i} \\ &= \{((a_1, \dots, a_n), f) \mid f : e_1 \times \dots \times e_n \rightarrow s \in F, a_i \in A_{e_i}, 1 \leq i \leq n\}. \end{aligned}$$

Hence for all  $s \in S$ ,  $k \in \mathbb{N}$  and  $t \in \mathcal{D}(k)$ ,

$$\mathcal{D}(0)_s = \emptyset,$$

$$\mathcal{D}(k+1)_s = H_\Sigma(\mathcal{D}(k))_s$$

$$= \{((t_1, \dots, t_n), f) \mid f : e_1 \times \dots \times e_n \rightarrow s \in F, t_i \in \mathcal{D}(k)_{e_i}, 1 \leq i \leq n\},$$

$$\mathcal{D}(k, k+1)(t) = t,$$

and thus by the Quotient Theorem,

$$\mu\Sigma_s = (\coprod_{k \in \mathbb{N}} \mathcal{D}(k)_s) / \sim_s \cong \bigcup_{k \in \mathbb{N}} \mathcal{D}(k)_s$$

where  $\sim_s$  is the equivalence closure of  $\{(t, \mathcal{D}(k, k+1)(t)) \mid t \in \mathcal{D}(k)_s, k \in \mathbb{N}\} = \Delta_{\mathcal{D}(k), s}$ .

By Lambek's Lemma (1),  $\alpha : H_\Sigma(A) \rightarrow A$  as defined in diagram (1) is iso and thus for all  $f : e_1 \times \dots \times e_n \rightarrow s \in F$  and  $t_i \in \mu\Sigma_{e_i}, 1 \leq i \leq n$ ,

$$f^{\mu\Sigma}(t_1, \dots, t_n) = ((t_1, \dots, t_n), f).$$

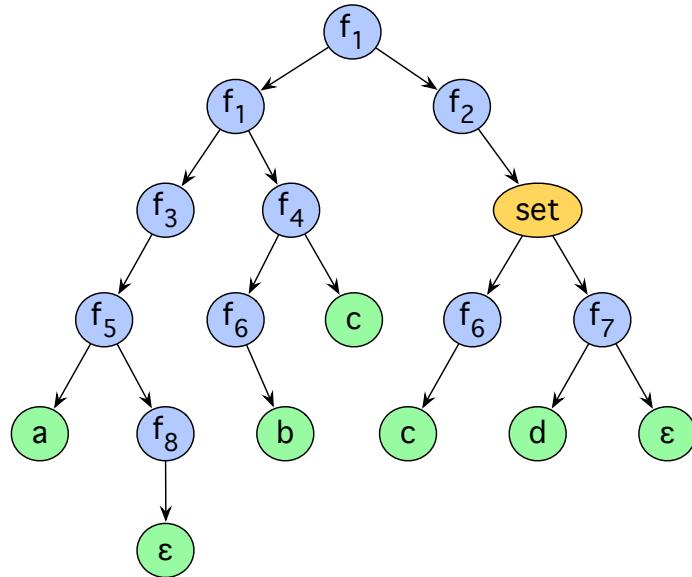
Hence for all  $\Sigma$ -algebras  $A$ ,

$$\text{fold}^{\mathcal{A}}(((t_1, \dots, t_n), f)) = \text{fold}^{\mathcal{A}}(f^{\mu\Sigma}(t_1, \dots, t_n)) = f^{\mathcal{A}}(\text{fold}_{e_1}^{\mathcal{A}}(t_1), \dots, \text{fold}_{e_n}^{\mathcal{A}}(t_n)).$$

Moreover, for  $A = \mu\Sigma$  and all  $s \in S$ ,

$$A_s \cong H_\Sigma(A)_s = \{((a_1, \dots, a_n), f) \mid f : e_1 \times \dots \times e_n \rightarrow s \in F, a_i \in A_{e_i}, 1 \leq i \leq n\}.$$

Hence  $\mu\Sigma$  can be represented as a quotient of  $T_\Sigma$  (see section 19.11: Term functors).



A ground  $\Sigma$ -term with constructors  $f_1, \dots, f_8$  and base elements  $a, b, c, d, \epsilon$ .

## Final models of destructive polynomial signatures

Let  $\Sigma = (S, \mathcal{I}, F)$  be a destructive polynomial signature,  $\kappa$  be the cardinality of the greatest exponent occurring in the range of some  $f \in F$  and  $\lambda$  be the first regular cardinal number  $> \kappa$ .

By Theorem **CONTYPES** (1) implies that  $H_\Sigma$  is  $\lambda$ -continuous and thus by Theorem **FINALG**,  $coAlg_{H_\Sigma}$  has a final object  $\alpha : \nu\Sigma \rightarrow H_\Sigma(\nu\Sigma)$ . In other words,  $\nu\Sigma$  is the final  $\Sigma$ -algebra (see (1)).

Since  $\nu\Sigma$  is the limit of the  $\omega$ -cochain  $\mathcal{D}$  of  $Set^S$  defined in Theorem **FINALG**, the **Subset Theorem** implies that for all  $s \in S$ ,

$$\nu\Sigma_s = \{a \in \prod_{i<\omega} \mathcal{D}(i)_s \mid \forall i < \omega : a_i = \mathcal{D}(i+1, i)(a_{i+1})\}.$$

Let  $A$  be a  $\Sigma$ -algebra. The unique  $\Sigma$ -homomorphism  $\text{unfold}^{\mathcal{A}} : A \rightarrow \nu\Sigma$  is the unique  $S$ -sorted function such that

$$A \xrightarrow{\langle \beta_i \rangle_{i < \omega}} \prod_{i < \omega} \mathcal{D}(i) = A \xrightarrow{\text{unfold}^{\mathcal{A}}} \nu\Sigma \xrightarrow{\text{inc}} \prod_{i < \omega} \mathcal{D}(i)$$

where  $\beta_0$  is the unique  $S$ -sorted function from  $A$  to  $\mathcal{D}(0)$  and for all  $i < \omega$  and  $s \in S$ ,

$$\beta_{i+1,s} = \langle F_e(\beta_{i,s}) \circ f^{\mathcal{A}} \rangle_{f:s \rightarrow e \in F} : A_s \rightarrow \mathcal{D}(i+1)_s.$$

$H_\Sigma$  is  $\omega$ -continuous and its object mapping reads as follows:

For all  $S$ -sorted sets  $A$  and  $s \in S$ ,

$$\begin{aligned} H_\Sigma(A)_s &= \prod_{f:s \rightarrow (e_1 + \dots + e_n)^X \in F} (\coprod_{i=1}^n A_{e_i})^X \\ &= \{t : F \rightarrow \bigcup_{f:s \rightarrow (e_1 + \dots + e_n)^X \in F} (\coprod_{i=1}^n A_{e_i})^X \mid \\ &\quad \forall f : s \rightarrow (e_1 + \dots + e_n)^X \in F : t(f) \in (\coprod_{i=1}^n A_{e_i})^X\} \\ &= \{\textcolor{red}{t} : F \rightarrow (A \times \mathbb{N})^X \mid \forall f : s \rightarrow (e_1 + \dots + e_n)^X \in F \forall x \in X \\ &\quad \exists 1 \leq i \leq n : \textcolor{red}{t}(f)(x) \in A_{e_i} \times \{i\}\}. \end{aligned}$$

Hence for all  $s \in S$ ,  $k \in \mathbb{N}$ ,  $t \in \mathcal{D}(k+1)$  and  $f \in F$ ,

$$\begin{aligned} \mathcal{D}(0)_s &= 1, \\ \mathcal{D}(k+1)_s &= H_\Sigma(\mathcal{D}(k))_s = \{t : F \rightarrow (\mathcal{D}(k) \times \mathbb{N})^X \mid \\ &\quad \forall f : s \rightarrow (e_1 + \cdots + e_n)^X \in F \ \forall x \in X \\ &\quad \exists 1 \leq i \leq n : t(f)(x) \in \mathcal{D}(k)_{e_i} \times \{i\}\}, \\ \mathcal{D}(k+1, k)(t)(f) &= \pi_1 \circ t(f), \end{aligned}$$

and thus by the **Subset Theorem**,

$$\begin{aligned} \nu\Sigma_s &= \{t \in \prod_{k \in \mathbb{N}} \mathcal{D}(k)_s \mid \forall k \in \mathbb{N} \ \forall f \in F : \mathcal{D}(k+1, k)(\pi_{k+1}(t))(f) = \pi_k(t)(f)\} \\ &= \{t \in \prod_{k \in \mathbb{N}} \mathcal{D}(k)_s \mid \forall k \in \mathbb{N} \ \forall f \in F : \pi_1 \circ \pi_{k+1}(t)(f) = \pi_k(t)(f)\}. \end{aligned}$$

## Bounded functors

Let  $\alpha : A \rightarrow F(A)$  be an  $F$ -coalgebra and  $B$  be a subset of  $A$ . If the inclusion mapping  $inc : B \rightarrow A$  is a  $coAlg_F$ -morphism from an  $F$ -coalgebra  $\beta : B \rightarrow F(B)$  to  $\alpha$  then  $\beta$  is an  **$F$ -invariant** or  **$F$ -subcoalgebra** of  $\alpha$ .

**Theorem** ([77], Prop. 6.2.4 (i)) Every union or intersection of  $F$ -invariants is an  $F$ -invariant. Hence for all subsets of  $B$  of  $A$  there is a least  $F$ -invariant  $\langle B \rangle : C \rightarrow F(C)$  such that  $C$  includes  $B$ . □

Let  $M$  be an  $S$ -sorted set.  $F : Set^S \rightarrow Set^S$  is  **$M$ -bounded** if for all  $F$ -coalgebras  $\alpha : A \rightarrow F(A)$  and  $a \in A$ ,  $|\langle a \rangle_s| \leq |M_s|$  (see [58], section 4).

This section aims at Theorem **BFIN**, which tells us that for all  $M$ -bounded functors  $F$ ,  $Alg_F$  has a final object.

Let  $\lambda$  be a cardinal number.

A category  $\mathcal{I}$  is  **$\lambda$ -filtered** if for each class  $\mathcal{L}$  of less than  $\lambda$   $\mathcal{I}$ -objects there is a cocone  $\{i \rightarrow j \mid i \in \mathcal{L}\}$  in  $\mathcal{I}$  and for all  $\mathcal{I}$ -objects  $i, j$  and each set  $\Phi$  of less than  $\lambda$   $\mathcal{I}$ -morphisms from  $i$  to  $j$  there is a coequalizing  $\mathcal{I}$ -morphism  $h : j \rightarrow k$ , i.e., for all  $f, g \in \Phi$ ,  $h \circ f = h \circ g$ .

A diagram  $\mathcal{D} : \mathcal{I} \rightarrow \mathcal{K}$  is  **$\lambda$ -filtered** if  $\mathcal{I}$  is a  $\lambda$ -filtered category.

A functor  $F : \mathcal{K} \rightarrow \mathcal{L}$  is  **$\lambda$ -accessible** if  $F$  preserves the colimits of all  $\lambda$ -filtered diagrams  $\mathcal{D} : \mathcal{I} \rightarrow \mathcal{K}$  (see [10], section 5.2).

**Theorem** ([11], Thm. 4.1; [12], 5.3)

Let  $M$  be an  $S$ -sorted set.  $F : Set^S \rightarrow Set^S$  is  $M$ -bounded if  $F$  is  $|M|$ -accessible. Conversely,  $F$  is  $(|M| + 1)$ -accessible if  $F$  is  $M$ -bounded.  $\square$

By [143], Thm. 10.6, or [58], Cor. 4.9, for every destructive signature  $\Sigma$  there is an  $S$ -sorted set  $M$  such that  $H_\Sigma$  is  $M$ -bounded (see  $\Sigma$ -functors).

## Examples

By [143], Ex. 6.8.2, or [58], Lemma 4.2,  $H_{DAut(X,Y)}$  is  $X^*$ -bounded:

For all  $DAut(X,Y)$ -algebras  $A$  and  $a \in A_{state}$ ,

$$\langle st \rangle = \{id_A^\#(a)(w), w \in X^*\}$$

(see State unfolding). Hence  $|\langle st \rangle| \leq |X^*|$ .

$B_{NAut(X,Y)}$  (see  $\Sigma$ -functors) is  $(X^* \times \mathbb{N})$ -bounded: For all  $B_{NAut(X,Y)}$ -algebras

$$A \xrightarrow{\langle \delta, \beta \rangle} B_{NAut(X,Y)}(A)$$

and  $a \in A_{state}$ ,

$$\langle st \rangle = \cup\{id_A^\#(a)(w), w \in X^*\}$$

where  $a \in A_{state}$ ,  $id_A^\#(a)(\epsilon) = \{st\}$  and  $id_A^\#(a)(x \cdot w) = \cup\{id_A^\#(st')(w) \mid st' \in \delta^A(a)(x)\}$  for all  $x \in X$  and  $w \in X^*$ . Since for all  $a \in A_{state}$  and  $x \in X$ ,  $|\delta^A(a)(x)| \in \mathbb{N}$ ,  $|\langle st \rangle| \leq |X^* \times \mathbb{N}|$ . If  $X = 1$ , then  $X^* \times \mathbb{N} \cong \mathbb{N}$  and thus  $B_{NAut(1,Y)}$  is  $\mathbb{N}$ -bounded (see [143], Ex. 6.8.1; [58], section 5.1).  $\square$

A destructive signature  $\Sigma = (S, \mathcal{I}, F)$  is **Moore-like** if there is an  $S$ -sorted set  $M$  such that for all  $f : s \rightarrow e \in F$ ,  $e = s^{M_s}$  or  $e \in \mathcal{I}$ . Then  $M$  is called the **input** of  $\Sigma$ .

## Lemma MOORE

Let  $\Sigma = (S, \mathcal{I}, F)$  be a Moore-like signature with input  $M$  and

$$F' = \{f : s \rightarrow e \mid e \in BT\}.$$

Let  $\kappa$  be the cardinality of the greatest exponent occurring in the source of some  $f \in F$  and  $\lambda$  be the first **regular cardinal number**  $> \kappa$ .

Since  $\Sigma$  is polynomial, Theorem **CONTYPES** (1) implies that  $H_\Sigma$  is  $\lambda$ -continuous and thus by Theorem **FINALG**,  $coAlg_{H_\Sigma}$  has a final object  $\alpha : A \rightarrow H_\Sigma(A)$ . In other words,  $Alg_\Sigma$  has a final object  $A$ .

Let  $Y = \prod_{f:s \rightarrow e \in F'} e$ . If  $|S| = 1$ , then  $\Sigma$  agrees with  $DAut(M_s, Y)$  and thus

$$A \cong Beh(M_s, Y).$$

Otherwise  $A$  can be constructed as a straightforward extension of  $Beh(M_s, Y)$  to several sorts: For all  $s \in S$  and  $h \in A_s$ ,

$$A_s = M_s^* \rightarrow Y,$$

for all  $f : s \rightarrow e \in F'$ ,  $f^{\mathcal{A}}(h) = \pi_g(h(\epsilon))$  and for all  $f : s \rightarrow s^{M_s}$ ,  $f^{\mathcal{A}}(h) = \lambda x. \lambda w. h(x \cdot w)$ .

$A$  can be visualized as the  $S$ -sorted set of trees such that for all  $s \in S$  and  $h \in A_s$ , the root  $r$  of  $h$  has  $|M_s|$  outarcs, for all  $f : s \rightarrow e \in F'$ ,  $r$  is labelled with  $f^{\mathcal{A}}(h)$ , and for all  $f : s \rightarrow s^{M_s}$  and  $x \in M_s$ ,  $f^{\mathcal{A}}(h)(x) = \lambda w. h(x \cdot w)$  is the subtree of  $h$  where the  $x$ -th outarc of  $r$  points to.  $\square$

## Theorem MOORETAU

Let  $\Sigma = (S, \mathcal{I}, F)$  be a destructive signature,  $M$  be an  $S$ -sorted set,  $H_{\Sigma}$  be  $M$ -bounded and

$$F' = \{f_s : s \rightarrow s^{M_s} \mid s \in S\} \cup \{f'_e : s \rightarrow M_e \mid f : s \rightarrow e \in F\}.$$

Of course,  $\Sigma' = (S, \mathcal{I}, F')$  is Moore-like.

Let the function  $\tau : H_{\Sigma'} \rightarrow H_{\Sigma}$  be defined as follows: For all  $S$ -sorted sets  $A$ ,  $a \in H_{\Sigma'}(A)_s$  and  $f : s \rightarrow e \in F$ ,

$$\pi_f(\tau_{A,s}(a)) = F_e(\pi_{f_s}(a))(\pi_{f'}(a)).$$

$\tau$  is a surjective natural transformation.

*Proof.* The theorem generalizes [58], Thm. 4.7 (i)  $\Rightarrow$  (iv), from  $Set$  to  $Set^S$ . □

## Theorem BFIN

Let  $\Sigma = (S, \mathcal{I}, F)$  be a destructive signature,  $M$  be an  $S$ -sorted set,  $H_{\Sigma}$  be  $M$ -bounded and the  $\Sigma$ -algebra  $A$  be defined as follows: For all  $s \in S$ ,

$$A_s = M_s^* \rightarrow \prod_{f:s \rightarrow e \in F} M_e,$$

and for all  $f : s \rightarrow e \in F$  and  $h \in A_s$ ,

$$f^{\mathcal{A}}(h) = F_e(\lambda x. \lambda w. h(x \cdot w))(\pi_f(h(\epsilon))).$$

$A$  is weakly final and  $A/\sim$  is final in  $Alg_{\Sigma}$  where  $\sim$  is the greatest  $\Sigma$ -congruence on  $A$ .

*Proof.* Let  $\Sigma'$  and  $\tau$  be defined as in Theorem MOORETAU. Let  $Y = \prod_{f':s \rightarrow M_e \in F'} M_e$ . Since  $\Sigma'$  is Moore-like, Lemma MOORE implies that the following  $\Sigma'$ -algebra  $B$  is final:

For all  $s \in S$ ,  $B_s = M_s^* \rightarrow Y$ .

For all  $f : s \rightarrow e \in F$  and  $h \in B_s$ ,  $f_s^B(h) = \lambda x. \lambda w. h(x \cdot w)$  and  $f'^B(h) = \pi_{f'}(h(\epsilon))$ .

Hence by Lemma WEAKFIN,  $A$  is weakly final:

For all  $s \in S$ ,  $\prod_{f:s \rightarrow e \in F} M_e = Y$  and thus  $A_s = B_s$ .

For all  $f : s \rightarrow e \in F$  and  $h \in A_s$ ,

$$\begin{aligned} f^A(h) &= F_e(\lambda x. \lambda w. h(x \cdot w))(\pi_f(h(\epsilon))) = F_e(\lambda x. \lambda w. h(x \cdot w))(\pi_{f'}(h(\epsilon))) \\ &= F_e(f_s^A(h))(f'^A(h)) = F_e(\pi_{f_s}(g_1(h), \dots, g_n(h)))(\pi_{f'}(g_1(h), \dots, g_n(h))) \\ &= \pi_f(\tau_{A,s}(g_1(h), \dots, g_n(h))) = \pi_f(\tau_{A,s}(\langle g_1, \dots, g_n \rangle(h))) = f^B(h) \end{aligned}$$

where  $\{g_1, \dots, g_n\} = \{g^A \mid g : s \rightarrow e' \in F'\}$ .

Hence again by Lemma WEAKFIN,  $A/\sim$  is final in  $Alg_\Sigma$  where  $\sim$  is the greatest  $\Sigma$ -congruence on  $A$ .

A direct proof of the existence of a final  $\Sigma$ -algebra is given by [59], Thm. 3.5. □

## Example

Let  $\Sigma = NAut(X, Y)$ , i.e.,  $S = \{state\}$ ,

$$F = \{\delta : state \rightarrow set(state)^X, \beta : state \rightarrow Y\}$$

and  $P = \emptyset$ , and  $M_{state} = X^* \times \mathbb{N}$ . Hence  $M_{set(state)^X} = \mathcal{P}_\omega(M)^X$  and  $M_Y = Y$ . Since  $H_\Sigma$  is  $M$ -bounded, Theorem BFIN implies that the following  $\Sigma$ -algebra  $A$  is weakly final:

$$A_{state} = M^* \rightarrow \mathcal{P}_\omega(M)^X \times Y.$$

For all  $h \in A_{state}$  and  $x \in X$ ,  $h(\epsilon) = (g, y)$  implies

$$\begin{aligned} \delta^A(h)(x) &= F_{set(state)^X}(\lambda m. \lambda w. h(mw))(\pi_\delta(h(\epsilon)))(x) \\ &= F_{set(state)^X}(\lambda m. \lambda w. h(mw))(g)(x) = F_{set(state)}(\lambda m. \lambda w. h(mw))(g(x)) \\ &= \{F_{state}(\lambda m. \lambda w. h(mw))(m) \mid m \in g(x)\} \\ &= \{\lambda m. \lambda w. h(mw))(m) \mid m \in g(x)\} = \{\lambda w. h(mw) \mid m \in g(x)\}, \\ \beta^A(h) &= F_Y(\lambda x. \lambda w. h(x \cdot w))(\pi_\beta(h(\epsilon))) = F_Y(\lambda x. \lambda w. h(x \cdot w))(y) = id_Y(y) = y. \end{aligned}$$

Moreover,  $A/\sim$  is final in  $Alg_\Sigma$  where  $\sim$  is the greatest  $\Sigma$ -congruence on  $A$ , i.e., the union of all  $S$ -sorted binary relations  $\sim$  on  $A$  such that for all  $h, h' \in A_{state}$ ,

$$h \sim h' \text{ implies } \delta^A(h) \sim_{set(state)^X} \delta^A(h') \wedge \beta^A(h) \sim_Y \beta^A(h'),$$

i.e., for all  $x \in X$ ,  $h \sim h'$ ,  $h(\epsilon) = (g, y)$  and  $h'(\epsilon) = (g', y')$  imply

$$\begin{aligned} \forall m \in g(x) \exists n \in g'(x) : \lambda w.h(mw) &\sim \lambda w.h'(nw) \wedge \\ \forall n \in g'(x) \exists m \in g(x) : \lambda w.h(mw) &\sim \lambda w.h'(nw) \wedge y = y'. \end{aligned}$$

Let  $F' = \{f : state \rightarrow state^M, \delta : state \rightarrow \mathcal{P}_\omega(M)^X, \beta : state \rightarrow Y\}$   
and  $\Sigma' = (S, \{X, Y, M, \mathcal{P}_\omega(M)^X\}, F')$ .

$A$  is constructed from the following  $\Sigma'$ -algebra  $B$  with  $B_{state} = A_{state}$  (see the proof of Theorem BFIN): For all  $h \in A_{state}$ ,  $f_{state}^B(h) = \lambda m. \lambda w. h(mw)$  and  $\langle \delta^B, \beta^B \rangle(h) = h(\epsilon)$ .

Since  $\Sigma'$  is Moore-like, Lemma MOORE implies that  $A$  can be visualized as the set of trees  $h$  such that the root  $r$  of  $h$  has  $|M|$  outarcs,  $r$  is labelled with  $h(\epsilon)$  and for all  $m \in M$ ,  $\lambda w.h(mw)$  is the subtree of  $h$  where the  $m$ -th outarc of  $r$  points to. [61], section 5, shows (for the case  $X = Y = 1$ ) how these trees yield the quotient  $A/\sim$ .  $\square$

## Adjunctions

An **adjunction** is a quadruple  $(L, R, \eta, \epsilon)$  consisting of functors  $L : \mathcal{K} \rightarrow \mathcal{L}$ ,  $R : \mathcal{L} \rightarrow \mathcal{K}$  and natural transformations  $\eta : Id_{\mathcal{K}} \rightarrow RL$  and  $\epsilon : LR \rightarrow Id_{\mathcal{L}}$  such that for every  $\mathcal{K}$ -morphism  $f : A \rightarrow RB$  there is a unique  $\mathcal{L}$ -morphism  $f^* : LA \rightarrow B$ , called the  **$\mathcal{K}$ -extension of  $f$** , such that the following diagram commutes:

$$\begin{array}{ccc}
 A & \xrightarrow{\eta_A} & RLA \\
 & \searrow f & \downarrow R(f^*) \\
 & & RB
 \end{array}
 \qquad
 \begin{array}{ccc}
 LA & & \\
 \downarrow & & \downarrow f^* \\
 B & & 
 \end{array}$$

or, equivalently, for every  $\mathcal{L}$ -morphism  $g : LA \rightarrow B$  there is a unique  $\mathcal{K}$ -morphism  $g^\# : A \rightarrow RB$ , called the  **$\mathcal{L}$ -coextension of  $g$** , such that the following diagram commutes:

$$\begin{array}{ccc}
 B & \xleftarrow{\epsilon_B} & LRB \\
 & \nwarrow g & \downarrow L(g^\#) \\
 & & LA
 \end{array}
 \qquad
 \begin{array}{ccc}
 RB & & \\
 \uparrow & & \uparrow g^\# \\
 A & & 
 \end{array}$$

$\eta$  is called a **unit** (or **inclusion of generators**).  $\epsilon$  called a **co-unit** (or **evaluation**).

An endofunctor  $T$  on  $\mathcal{K}$  is called **pointed** if there is a natural transformation  $\eta : Id_{\mathcal{K}} \rightarrow T$  [70]. Hence  $RL$  is pointed.

Since  $\eta$  is a natural transformation, for all  $g \in \mathcal{K}(A, C)$ ,  $RL(g) \circ \eta_A = \eta_C \circ g$ . Hence by the uniqueness of  $\mathcal{K}$ -extensions,  $Lg = (\eta_C \circ g)^*$  and thus for all  $B \in \mathcal{L}$  and  $f \in \mathcal{K}(A, RB)$ ,

$$\begin{aligned} R(id_{RB}^* \circ Lf) \circ \eta_A &= R(id_{RB}^*) \circ RLf \circ \eta_A = R(id_{RB}^*) \circ R((\eta_{RB} \circ f)^*) \circ \eta_A \\ &= R(id_{RB}^*) \circ \eta_{RB} \circ f = id_{RB} \circ f = f. \end{aligned} \tag{1}$$

Again by the uniqueness of  $\mathcal{K}$ -extensions, (1) implies  $id_{RB}^* \circ Lf = f^*$ . (2)

Since  $\epsilon$  is a natural transformation, for all  $g \in \mathcal{L}(C, B)$ ,  $\epsilon_B \circ LR(g) = g \circ \epsilon_C$ . Hence by the uniqueness of  $\mathcal{L}$ -coextensions,  $Rg = (g \circ \epsilon_C)^\#$  and thus for all  $A \in \mathcal{K}$  and  $f \in \mathcal{L}(LA, B)$ ,

$$\begin{aligned} \epsilon_B \circ L(Rf \circ id_{LA}^\#) &= \epsilon_B \circ L(Rf) \circ L(id_{LA}^\#) = \epsilon_B \circ L((f \circ \epsilon_{LA})^\#) \circ L(id_{LA}^\#) \\ &= f \circ \epsilon_{LA} \circ L(id_{LA}^\#) = f \circ id_{LA} = f. \end{aligned} \tag{3}$$

Again by the uniqueness of  $\mathcal{L}$ -coextensions, (2) implies  $Rf \circ id_{LA}^\# = f^\#$ . (4)

$L$  is called the **left adjoint** of  $R$  and  $R$  the **right adjoint** of  $L$ . We write  $L \dashv R$ .

For all  $A \in \mathcal{K}$ ,  $L(A)$  is called **free over**  $A$ .

For all  $B \in \mathcal{L}$ ,  $R(B)$  is called **cofree over**  $B$ .

Left adjoints preserve colimits and thus initial objects.

Right adjoints preserve limits and thus final objects.

$L \dashv R$  holds true iff the hom-functors

$$\mathcal{K}(\_, R(\_)), \mathcal{L}(L(\_), \_) : \mathcal{K}^{op} \times \mathcal{L} \rightarrow Set$$

are naturally equivalent,

i.e., for all  $A, A' \in \mathcal{K}$ ,  $f \in \mathcal{K}(A', A)$ ,  $B, B' \in \mathcal{L}$  and  $g \in \mathcal{L}(B, B')$ ,

$$\mathcal{K}(A, RB) \cong \mathcal{L}(LA, B) \tag{5}$$

and the following diagram commutes (see chapter 5):

$$\begin{array}{ccc}
 \mathcal{K}(A, RB) & \cong & \mathcal{L}(LA, B) \\
 \downarrow \mathcal{K}(f, Rg) = \lambda h. Rg \circ h \circ f & & \downarrow \mathcal{L}(Lf, g) = \lambda h. g \circ h \circ Lf \\
 \mathcal{K}(A', RB') & \cong & \mathcal{L}(LA', B')
 \end{array}$$

Let  $L \dashv R$ . Then the bijection (3) is defined as follows:

For all  $f \in \mathcal{K}(A, RB)$  and  $g \in \mathcal{L}(LA, B)$ ,

$$\phi(f) = \epsilon_B \circ Lf \quad \text{and} \quad \phi^{-1}(g) = Rg \circ \eta_A. \quad (6)$$

(5) also motivates the following rule notations of an adjunction:

$$\frac{\begin{array}{c} A \xrightarrow{f} RB \\ \hline LA \xrightarrow{f^*} B \end{array}}{\frac{LA \xrightarrow{g} B}{A \xrightarrow{g^\#} RB}}$$

(6) implies

$$\begin{aligned} id_{RB} &= \phi^{-1}(\phi(id_{RB})) = \phi^{-1}(\epsilon_B \circ Lid_{RB}) = \phi^{-1}(\epsilon_B \circ id_{LRB}) = \phi^{-1}(\epsilon_B) \\ &= R\epsilon_B \circ \eta_{RB}, \end{aligned} \quad (7)$$

$$id_{LA} = \phi(\phi^{-1}(id_{LA})) = \phi(Rid_{LA} \circ \eta_A) = \phi(id_{RLA} \circ \eta_A) = \phi(\eta_A) = \epsilon_B \circ L\eta_A. \quad (8)$$

By the uniqueness of  $\mathcal{K}$ -extensions, (7) implies  $\epsilon_B = id_{RB}^*$ .

By the uniqueness of  $\mathcal{L}$ -coextensions, (8) implies  $\eta_A = id_{LA}^\#$ .

## Sample adjunctions

1. The identity functor  $\text{Id}_{\mathcal{K}} : \mathcal{K} \rightarrow \mathcal{K}$  is left and right adjoint to itself.

$$\frac{A \xrightarrow{f} \text{Id}_{\mathcal{K}}(B)}{A \xrightarrow{f^*=f} B} \quad \frac{\text{Id}_{\mathcal{K}}(A) \xrightarrow{g} B}{A \xrightarrow{g^\#=g} B}$$

2. Let  $\text{Monoid}$  be the full subcategory of  $\text{Alg}_{\text{Mon}}$  whose objects are monoids.

The **monoid functor** (see chapter 8; [16], section 7.2)

$$MF : \text{Set} \rightarrow \text{Monoid} \subseteq \text{Alg}_{\text{Mon}}$$

$$X \mapsto (X^*, \{\lambda \epsilon. \epsilon : 1 \rightarrow X^*, \lambda(v, w).vw : (X^*)^2 \rightarrow X^*\})$$

$$f : X \rightarrow A \mapsto MF(f) : X^* \rightarrow A^*$$

where for all  $x \in X$  and  $w \in X^*$ ,

$$MF(f)(\epsilon) =_{\text{def}} \epsilon,$$

$$MF(f)(x \cdot w) =_{\text{def}} f(x) \cdot MF(f)(w),$$

is left adjoint to the forgetful functor  $U : \text{Monoid} \rightarrow \text{Set}$  (which maps a monoid to its carrier), i.e., for all monoids  $\mathcal{A}$  and functions  $f : X \rightarrow U(\mathcal{A})$  there is a unique  $\text{Mon}$ -homomorphism  $f^* : MF(X) \rightarrow \mathcal{A}$  such that (1) commutes:

$$\begin{array}{ccc}
 X & \xrightarrow{\eta_X =_{\text{def}} inc_X} & X^* \\
 & \searrow f & \downarrow U(f^*) \\
 & & U(\mathcal{A})
 \end{array}
 \quad
 \begin{array}{cc}
 MF(X) & \text{free monoid over } X \\
 & \downarrow f^* \\
 & \mathcal{A}
 \end{array}$$

For all  $x \in X$  and  $w \in X^*$ ,

$$\begin{aligned}
 f^*(\epsilon) &=_{\text{def}} \text{one}^{\mathcal{A}}, \\
 f^*(x \cdot w) &=_{\text{def}} \text{mul}^{\mathcal{A}}(f(x), f^*(w)).
 \end{aligned}$$

Equivalently, for all monoids  $\mathcal{A}$  and  $\text{Mon}$ -homomorphisms  $g : MF(X) \rightarrow \mathcal{A}$  there is a unique function  $g^\# : X \rightarrow U(\mathcal{A})$  such that (2) commutes:

$$\begin{array}{ccc}
 \mathcal{A} & \xleftarrow{\epsilon_{\mathcal{A}}} & MF(U(\mathcal{A})) \\
 & \swarrow g & \downarrow \text{MF}(g^\#) \\
 & & MF(X)
 \end{array}
 \quad
 \begin{array}{ccc}
 U(\mathcal{A}) & \xrightarrow{\quad} & X \\
 & \downarrow & \downarrow g^\# =_{\text{def}} \lambda x.g(x)
 \end{array}$$

For all  $a \in U(\mathcal{A})$  and  $w \in U(\mathcal{A})^*$ ,

$$\begin{aligned}\epsilon_{\mathcal{A}}(\epsilon) &=_{def} \text{one}^{\mathcal{A}}, \\ \epsilon_{\mathcal{A}}(aw) &=_{def} \text{mul}^{\mathcal{A}}(a, \epsilon_{\mathcal{A}}(w)).\end{aligned}$$

Summing up:

$$\frac{X \xrightarrow{f} U(\mathcal{A})}{MF(X) \xrightarrow{f^*} \mathcal{A}} \quad \frac{MF(X) \xrightarrow{g} \mathcal{A}}{X \xrightarrow{g^\#} U(\mathcal{A})}$$

### 3. The sequence functor (see sample algebras 3; [16], section 7.2)

$$\begin{aligned}SF_X : \mathbf{Set} &\rightarrow \mathbf{Alg}_{coStream(X)} \\ Y &\mapsto (X^* \times Y, \{cons^{Seq(X,Y)}\}) \\ f : Y \rightarrow A &\mapsto \lambda(w, y).(w, f(y)) : X^* \times Y \rightarrow X^* \times A\end{aligned}$$

is left adjoint to the forgetful functor  $\mathbf{U} : \mathbf{Alg}_{coStream(X)} \rightarrow \mathbf{Set}$ , i.e., for all  $coStream(X)$ -algebras  $\mathcal{A}$  and functions  $f : Y \rightarrow U(\mathcal{A})$  there is a unique  $coStream(X)$ -homomorphism  $f^* : SF_X(Y) \rightarrow \mathcal{A}$  such that (3) commutes:

$$\begin{array}{ccc}
 Y & \xrightarrow{\eta_Y = \lambda y.(\epsilon, y)} & X^* \times Y \\
 & \searrow f & \downarrow U(f^*) \\
 & & U(\mathcal{A})
 \end{array}$$

free  $\text{coStream}(X)$ -algebra over  $Y$   
= initial  $\text{Dyn}(X, Y)$ -algebra  
with  $\alpha^{SF_X(Y)} = \eta_Y$

$\text{coStream}(X)$ -algebra  
=  $\text{Dyn}(X, Y)$ -algebra with  $\alpha^{\mathcal{A}} = f$

For all  $y \in Y$ ,  $x \in X$  and  $w \in X^*$ ,

$$\begin{aligned}
 f^*(\epsilon, y) &=_{\text{def}} f(y), \\
 f^*(xw, y) &=_{\text{def}} \text{cons}^{\mathcal{A}}(x, f^*(w, y)).
 \end{aligned}$$

Equivalently, for all  $\text{coStream}(X)$ -algebras  $\mathcal{A}$  and  $\text{coStream}(X)$ -homomorphisms  $g : SF_X(Y) \rightarrow U(\mathcal{A})$  there is a unique function  $g^\# : Y \rightarrow U(\mathcal{A})$  such that (4) commutes:

$$\begin{array}{ccc}
 \mathcal{A} & \xleftarrow{\epsilon_{\mathcal{A}}} & SF_X(U(\mathcal{A})) \\
 & \nwarrow g & \downarrow \lambda \\
 & & SF_X(g^\#) \\
 & \searrow & \downarrow \\
 & & SF_X(Y)
 \end{array}
 \qquad
 \begin{array}{c}
 U(\mathcal{A}) \\
 \downarrow \lambda \\
 g^\# =_{\text{def}} \lambda y.g(\epsilon, y) \\
 \downarrow \\
 Y
 \end{array}$$

For all  $a \in U(\mathcal{A})$ ,  $x \in X$  and  $w \in X^*$ ,

$$\begin{aligned}\epsilon_{\mathcal{A}}(\epsilon, a) &=_{def} a, \\ \epsilon_{\mathcal{A}}(xw, a) &=_{def} \text{cons}^{\mathcal{A}}(x, \epsilon_{\mathcal{A}}(w, a)).\end{aligned}$$

Summing up:

$$\frac{Y \xrightarrow{f} U(\mathcal{A})}{SF_X(Y) \xrightarrow{f^*} \mathcal{A}} \qquad \frac{SF_X(Y) \xrightarrow{g} \mathcal{A}}{Y \xrightarrow{g^\#} U(\mathcal{A})}$$

4. The **behavior functor** (see sample algebra 24; [16], section 7.2; [17], section 3.1)

$$\begin{aligned}BF_X : Set &\rightarrow Alg_{Med(X)} \\ Y &\mapsto (Y^{X^*}, \{\delta^{Beh(X,Y)}\}) \\ g : A \rightarrow Y &\mapsto \lambda h.g \circ h : A^{X^*} \rightarrow Y^{X^*}\end{aligned}$$

is right adjoint to the forgetful functor  $U : Alg_{Med(X)} \rightarrow Set$ , i.e., for all  $Med(X)$ -algebras  $\mathcal{A}$  and functions  $g : U(\mathcal{A}) \rightarrow Y$  there is a unique  $Med(X)$ -homomorphism  $g^\# : \mathcal{A} \rightarrow BF_X(Y)$  such that (5) commutes:

$$\begin{array}{ccc}
 Y & \xleftarrow{\epsilon_Y = \lambda h.h(\epsilon)} & Y^{X^*} \\
 & \nwarrow g & \downarrow U(g^\#) \\
 & (5) & \\
 & & U(\mathcal{A}) \\
 & & \downarrow \\
 & & \mathcal{A}
 \end{array}$$

cofree  $Med(X)$ -algebra over  $Y$   
 = final  $DAut(X, Y)$ -algebra  
 with  $\beta^{BF_X(Y)} = \epsilon_Y$   
  
 $Med(X)$ -algebra  
 $= DAut(X, Y)$ -algebra with  $\beta^{\mathcal{A}} = g$

For all  $a \in U(\mathcal{A})$ ,  $x \in X$  and  $w \in X^*$ ,

$$\begin{aligned}
 g^\#(a)(\epsilon) &=_{def} g(a), \\
 g^\#(a)(x \cdot w) &=_{def} g^\#(\delta^{\mathcal{A}}(a)(x))(w).
 \end{aligned}$$

Equivalently, for all  $Med(X)$ -algebras  $\mathcal{A}$  and  $Med(X)$ -homomorphisms  $f : \mathcal{A} \rightarrow BF_X(Y)$  there is a unique function  $f^* : U(\mathcal{A}) \rightarrow Y$  such that (6) commutes:

$$\begin{array}{ccc}
 \mathcal{A} & \xrightarrow{\eta_{\mathcal{A}}} & BF_X(U(\mathcal{A})) \\
 & \searrow f & \downarrow \\
 & (6) & \\
 & & BF_X(f^*) \\
 & & \downarrow \\
 & & BF_X(Y)
 \end{array}
 \quad
 \begin{array}{c}
 U(\mathcal{A}) \\
 \downarrow \\
 f^* =_{def} \lambda a.f(a)(\epsilon) \\
 \downarrow \\
 Y
 \end{array}$$

For all  $a \in U(\mathcal{A})$ ,  $x \in X$  and  $w \in X^*$ ,

$$\begin{aligned}\eta_{\mathcal{A}}(a)(\epsilon) &=_{def} a, \\ \eta_{\mathcal{A}}(a)(x \cdot w) &=_{def} \eta_{\mathcal{A}}(\delta^{\mathcal{A}}(a)(x))(w).\end{aligned}$$

Summing up:

$$\frac{\mathcal{A} \xrightarrow{f} BF_X(Y)}{U(\mathcal{A}) \xrightarrow{f^*} Y} \qquad \frac{U(\mathcal{A}) \xrightarrow{g} Y}{\mathcal{A} \xrightarrow{g^\#} BF_X(Y)}$$

**5.** Let  $(R, +, 0, *, 1)$  be a semiring (see sample algebra 25).

The **weighted-set functor**  $R_{\omega}^- : \text{Set} \rightarrow \text{SMod}_R$  is left adjoint to the forgetful functor  $U : \text{SMod}_R \rightarrow \text{Set}$ .

For all  $R$ -semimodules  $A$  with  $R$ -action  $\cdot : R \times A \rightarrow A$  and functions  $f : X \rightarrow A$  there is a unique linear function  $f^* : R_{\omega}^X \rightarrow A$  such that (7) commutes:

For all  $g : X \rightarrow_{\omega} R$ ,  $f^*(g) =_{def} \sum_{x \in \text{supp}(g)} g(x) \cdot f(x)$ .

$$\begin{array}{ccc}
 X & \xrightarrow{\lambda x.(\lambda y.0)[1/x]} & R_\omega^X \\
 & \searrow f & \downarrow f^* \\
 & (7) & \downarrow \gamma \\
 & & A
 \end{array}$$

free  $R$ -semimodule over  $X$

$R$ -semimodule

Particular cases of this adjunction have a **ring** (= semiring with additive inverses) or a **field** (= ring with commutative multiplication and multiplicative inverses) instead of a semiring and thus provide the free  $R$ -module (=  $R$ -semimodule with ring  $R$ ) or the free  $R$ -vector space (=  $R$ -module with field  $R$ ) over  $X$ .

Linearization (also called determinization) of weighted automata

Let  $\mathcal{A}$  be an  $R$ -weighted automaton with carrier  $A$  and output in  $R$ , i.e., a  $WAut(X, R, R)$ -algebra. The linearization of  $\mathcal{A}$ ,  $Lin(\mathcal{A})$ , is the linear automaton, i.e., the  $DAut(X, R)$ -algebra that is defined as follows:

$$\begin{aligned}
 Lin(\mathcal{A})_{state} &= R_\omega^A, \\
 \delta^{Lin(\mathcal{A})} &= (\delta^{\mathcal{A}})^*: R_\omega^A \rightarrow (R_\omega^A)^X, \\
 \beta^{Lin(\mathcal{A})} &= (\beta^{\mathcal{A}})^*: R_\omega^A \rightarrow R.
 \end{aligned}$$

6. Let  $A, B$  be sets,  $f : A \rightarrow B$  and  $\mathcal{P}(A), \mathcal{P}(B)$  be the categories with subsets of  $A, B$ , respectively, as objects and set inclusions as morphisms. The pre-images of  $f$  yield the functor

$$f^{-1} : \mathcal{P}(B) \rightarrow \mathcal{P}(A)$$

$$Y \mapsto \{a \in A \mid f(a) \in Y\}.$$

The following functors are left or right adjoint to  $f^{-1}$ :

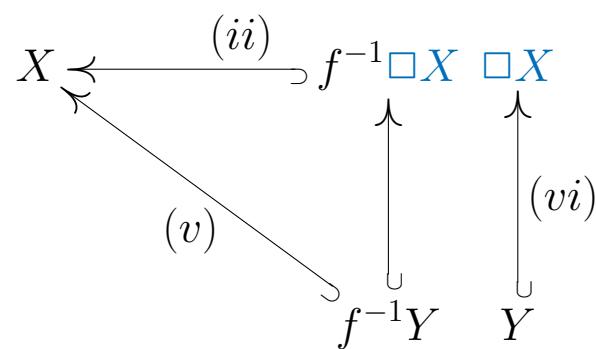
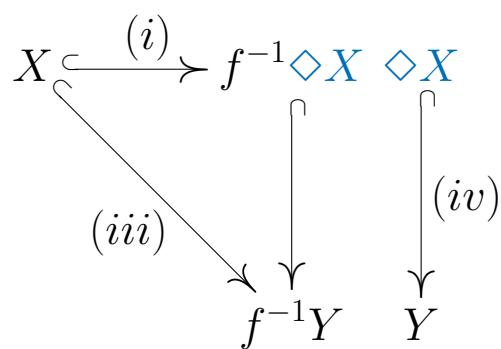
$$\diamond : \mathcal{P}(A) \rightarrow \mathcal{P}(B)$$

$$X \mapsto \{b \in B \mid f^{-1}(b) \cap X \neq \emptyset\}$$

$$= f(X) = \{f(x) \mid x \in X\}$$

$$\square : \mathcal{P}(A) \rightarrow \mathcal{P}(B)$$

$$X \mapsto \{b \in B \mid f^{-1}(b) \subseteq X\}$$



*Proof.*

$$(i) \quad X \subseteq \{a \in A \mid f(a) \in f(X) = \diamond X\} = f^{-1}\diamond X$$

$$(ii) \quad f^{-1}\square X = \{a \in A \mid f(a) \in \square X = \{b \in B \mid f^{-1}(b) \subseteq X\}\}$$

$$= \{a \in A \mid f^{-1}(f(a)) \subseteq X\} = \{a \in A \mid \{a' \in A \mid f(a') = f(a)\} \subseteq X\}$$

$$= \{a \in A \mid \forall a' \in A : f(a') = f(a) \Rightarrow a' \in X\} \subseteq X$$

$$(iii) \quad X \subseteq f^{-1}Y$$

$$\Rightarrow \diamond X = \{b \in B \mid f^{-1}(b) \cap X \neq \emptyset\} \subseteq \{b \in B \mid f^{-1}(b) \cap f^{-1}Y \neq \emptyset\}$$

$$\Leftrightarrow \diamond X \subseteq \{b \in B \mid \{a \in A \mid f(a) = b\} \cap \{a \in A \mid f(a) \in Y\} \neq \emptyset\}$$

$$\Leftrightarrow \diamond X \subseteq \{b \in B \mid \{a \in A \mid f(a) = b \in Y\} \neq \emptyset\}$$

$$\Leftrightarrow (iv) \quad \diamond X \subseteq \{b \in B \mid \{a \in A \mid f(a) = b \in Y\} \neq \emptyset\} \subseteq Y$$

$$(v) \quad f^{-1}Y \subseteq X$$

$$\Rightarrow \{b \in B \mid f^{-1}(b) \subseteq f^{-1}Y\} \subseteq \{b \in B \mid f^{-1}(b) \subseteq X\} = \square X$$

$$\Leftrightarrow \{b \in B \mid \{a \in A \mid f(a) = b\} \subseteq \{a \in A \mid f(a) \in Y\}\} \subseteq \square X$$

$$\Leftrightarrow \{b \in B \mid \forall a \in A : (f(a) = b \Rightarrow f(a) \in Y)\} \subseteq \square X$$

$$\Leftrightarrow (vi) \quad Y \subseteq \{b \in B \mid \forall a \in A : (f(a) = b \Rightarrow f(a) \in Y)\} \subseteq \square X$$

7.

Let  $G \in \text{Alg}_{\text{Graph}}$  (see chapter 8).  $(e_0, \dots, e_{n-1}) \in G_{\text{edge}}^+$  is a **cycle** of  $G$  if for all  $0 \leq i < n$ ,  $\text{target}^G(e_i) = \text{source}^G(e_{(i+1) \bmod n})$ .  $\text{cycles}(G)$  denotes the set of cycles of  $G$ .

Let  $S = \{\text{node}, \text{edge}\}$ . The  $S$ -sorted equivalence relation  $\sim$  relates all strongly connected nodes of  $G$  to each other: For all  $e, e' \in G_{\text{edge}}$  and  $a, b \in G_{\text{node}}$ ,

$$\begin{aligned} a \sim_{\text{node}} b &\Leftrightarrow_{\text{def}} a = b \vee \left\{ \begin{array}{l} \exists (e_0, \dots, e_{n-1}) \in \text{cycles}(G), 0 \leq i, j < n : \\ a = \text{source}^G(e_i) \wedge b = \text{source}^G(e_j), \end{array} \right. \\ e \sim_{\text{edge}} e' &\Leftrightarrow_{\text{def}} e = e' \vee \exists (e_0, \dots, e_{n-1}) \in \text{cycles}(G), 0 \leq i, j < n : e = e_i \wedge e' = e_j. \end{aligned}$$

$\sim_G = (\sim_{\text{node}}, \sim_{\text{edge}})$  is a *Graph*-congruence, i.e., for all  $e, e' \in G_{\text{edge}}$ ,  $e \sim_{\text{edge}} e'$  implies  $\text{source}^G(e) \sim_{\text{node}} \text{source}^G(e')$  and  $\text{target}^G(e) \sim_{\text{node}} \text{target}^G(e')$ .

Let *Acyclic* be the full subcategory of  $\text{Alg}_{\text{Graph}}$  whose objects are the acyclic graphs.

The functor  $\text{SC} : \text{Alg}_{\text{Graph}} \rightarrow \text{Acyclic}$  that maps each graph  $G$  to the acyclic graph  $G/\sim_G$  of the strongly connected components of  $G$  is left adjoint to the inclusion functor  $\text{Inc} : \text{Acyclic} \rightarrow \text{Alg}_{\text{Graph}}$  ([133], Ex. 2.4.10).

$$\begin{array}{ccc}
 G & \xrightarrow{\text{nat} \sim} & \text{Inc}(\text{SC}(G)) \\
 & \searrow f & \downarrow \\
 & & \text{Inc}(f^*) \\
 & & \downarrow \\
 & & \text{Inc}(A) \\
 & & \downarrow \\
 & & A
 \end{array}
 \quad \begin{array}{c}
 \text{SC}(G) = G/\sim \\
 f^* = \lambda[e]_\sim.f(e) \\
 \text{acyclic graph}
 \end{array}$$

$f^*$  is well-defined: Let  $e \sim_{\text{edge}} e'$ . If  $f_{\text{edge}}(e) \neq f_{\text{edge}}(e')$ , then  $e \neq e'$  and thus  $g$  has a cycle  $(e_0, \dots, e_{n-1})$ . Since  $f$  is a *Graph*-homomorphic, for all  $0 \leq i < n$ ,

$$\begin{aligned}
 \text{target}^{\mathcal{A}}(f_{\text{edge}}(e_i)) &= f_{\text{edge}}(\text{target}^G(e_i)) = f_{\text{edge}}(\text{source}^G(e_{(i+1) \bmod n})) \\
 &= \text{source}^{\mathcal{A}}(f_{\text{edge}}(e_{(i+1) \bmod n})).
 \end{aligned}$$

Hence  $(f_{\text{edge}}(e_0), \dots, f_{\text{edge}}(e_{n-1}))$  is a cycle of the acyclic graph  $A$ .  $\sharp$

We conclude  $f_{\text{edge}}(e) = f_{\text{edge}}(e')$ .

Let  $a \sim_{\text{node}} b$ . If  $f_{\text{node}}(a) \neq f_{\text{node}}(b)$ , then  $a \neq b$  and thus  $G$  has a cycle. As above we obtain a contradiction and thus conclude  $f_{\text{node}}(a) = f_{\text{node}}(b)$ .

Since  $\text{nat}_{\sim_G}$  and  $f = f^* \circ \text{nat}_{\sim_G}$  are graph homomorphisms and  $\text{nat}_{\sim_G}$  is surjective, Lemma EMH (1) implies that  $f^*$  is also a graph homomorphism. Hence the uniqueness of  $f^*$  satisfying  $f = f^* \circ \text{nat}_{\sim_G}$  again follows from the fact that  $\text{nat}_{\sim_G}$  is epi.

8. Reader functors are left adjoint to writer functors.

$$\frac{A \xrightarrow{f} C^B}{A \times B \xrightarrow{f^*} C}$$

$$\frac{A \times B \xrightarrow{g} C}{A \xrightarrow{g^\#} C^B}$$

In Haskell,  $f^*$  and  $g^\#$  are called *uncurry(f)* and *curry(g)*, respectively.

$$\begin{array}{ccc}
 A & \xrightarrow{\eta_A} & (A \times B)^B \\
 & \searrow f & \swarrow (f^*)^B \\
 & C^B &
 \end{array}
 \qquad
 \begin{array}{ccc}
 C & \xleftarrow{\epsilon_C} & C^B \times B \\
 & \nwarrow g & \swarrow g^\# \times B \\
 & A \times B &
 \end{array}$$

This example suggests a notion of adjoint signatures  $\Sigma$  and  $\Sigma'$  with the same set of sorts:  $\Sigma$  is **left (right) adjoint to**  $\Sigma'$  if  $H_\Sigma$  is left (right) adjoint to  $H_{\Sigma'}$  (in  $Mod(S, \mathcal{I})$ ; see chapter 13).

For instance,  $Med(X)$  is left adjoint to  $coStream(X)$  (see chapter 8).

Indeed, for all  $Med(X)$ -algebras  $\mathcal{A}$  and  $coStream(X)$ -algebras  $\mathcal{B}$ , both with carrier  $A$ , interpretations of  $\delta : state \times X \rightarrow state$  in  $\mathcal{A}$  and  $\delta : state \rightarrow state^X$  in  $\mathcal{B}$  are obtained as the extension of  $\delta^{\mathcal{A}}$  and the coextension of  $\delta^{\mathcal{B}}$ , respectively:

$$\frac{A \xrightarrow{\delta^{\mathcal{A}}} A^X}{A \times X \xrightarrow{(\delta^{\mathcal{A}})^*} A} \quad \frac{A \times X \xrightarrow{\delta^{\mathcal{B}}} A}{A \xrightarrow{(\delta^{\mathcal{B}})^\#} A^X}$$

A category  $\mathcal{K}$  is **Cartesian closed** if  $\mathcal{K}$  has a final object, binary products and for all  $B \in \mathcal{K}$  there is an adjunction  $(L : \mathcal{K} \rightarrow \mathcal{K}, R : \mathcal{K} \rightarrow \mathcal{K}, \eta, \epsilon)$  such that for all  $A \in \mathcal{K}$  and  $\mathcal{K}$ -morphisms  $f : A \rightarrow C$ ,  $L(A) = A \times B$ ,  $L(f) = f \times B$ ,  $R(A) = A^B$  and  $R(f) = f^B$ .

$Set^S$  is Cartesian closed: Let  $A, B, C$  be  $S$ -sorted sets.

- $C^B = Set^S(B, C)$ .
- For all  $S$ -sorted functions  $f : A \rightarrow C$  and  $g : B \rightarrow A$ ,  $f^B(g) =_{def} f \circ g$ .
- $\epsilon_C = apply =_{def} \lambda(f, b).f(b)$  and  $\eta_A = pair =_{def} \lambda a. \lambda b.(a, b)$ .
- For all  $S$ -sorted functions  $g : A \times B \rightarrow C$ ,  $g^\# =_{def} \lambda a. \lambda b. g(a, b)$ .
- For all  $S$ -sorted functions  $h : A \rightarrow C^B$ ,  $h^* =_{def} \lambda(a, b). h(a)(b)$ .

9. Products (and other limits) are right adjoint to diagonals.

$$\frac{A \xrightarrow{f} B \times C}{(A, A) \xrightarrow{f^* = (\pi_1 \circ f, \pi_2 \circ f)} (B, C)}$$

$$\frac{(A, A) \xrightarrow{(f, g)} (B, C)}{A \xrightarrow{(f, g)\# = \langle f, g \rangle} B \times C}$$

$$\begin{array}{ccc}
 A & \xrightarrow{\eta_A = \langle id_A, id_A \rangle} & A \times A \\
 & \searrow f & \downarrow (\pi_1 \circ f) \times (\pi_2 \circ f) \\
 & & B \times C
 \end{array}
 \qquad
 \begin{array}{ccc}
 (A, A) & & \\
 \downarrow & & \downarrow (\pi_1 \circ f, \pi_2 \circ f) \\
 & & (B, C)
 \end{array}$$

$$\begin{array}{ccc}
 (B, C) & \xleftarrow{\epsilon_{(B,C)} = (\pi_1, \pi_2)} & (B \times C, B \times C) \\
 & \nwarrow (f, g) & \downarrow \lambda \\
 & & (\langle f, g \rangle, \langle f, g \rangle) \\
 & & \downarrow \\
 & & (A, A)
 \end{array}
 \qquad
 \begin{array}{ccc}
 B \times C & & \\
 \downarrow \lambda & & \downarrow \langle f, g \rangle \\
 & & A
 \end{array}$$

$R : \mathcal{K}^I \rightarrow \mathcal{K}$  with  $R((B_i)_{i \in I}) = \prod_{i \in I} B_i$  for all  $\mathcal{K}^I$ -objects and -morphisms  $(B_i)_{i \in I}$  is right adjoint to the diagonal functor  $\Delta_{\mathcal{K}}^I : \mathcal{K} \rightarrow \mathcal{K}^I$  (see [Functors](#)).

$$\begin{array}{ccc}
 A & \xrightarrow{\eta_A = \langle id_A \rangle_{i \in I}} & A^I \\
 & \searrow f & \downarrow \prod_{i \in I} (\pi_i \circ f) \\
 & & \prod_{i \in I} B_i \\
 & & \uparrow \epsilon_{(B_i)_{i \in I}} = (\pi_i)_{i \in I} \\
 (B_i)_{i \in I} & \xleftarrow{(g_i)_{i \in I}} & \prod_{i \in I} B_i \\
 & \swarrow & \downarrow \lambda (\langle g_i \rangle_{i \in I})_{i \in I} \\
 & & (A)_{i \in I}
 \end{array}$$

10. Coproducts (and other colimits) are left adjoint to diagonals.

$$\frac{(A, B) \xrightarrow{(f,g)} (C, C)}{A + B \xrightarrow{(f,g)^* = [f,g]} C} \quad \frac{A + B \xrightarrow{f} C}{(A, B) \xrightarrow{f\# = (f \circ \iota_1, f \circ \iota_2)} (C, C)}$$

$$(A, B) \xrightarrow{\eta_{(A,B)} = (\iota_1, \iota_2)} (A + B, A + B)$$

↓

$$(f, g) \searrow \begin{matrix} ([f, g], [f, g]) \\ \downarrow \\ (C, C) \end{matrix}$$

↓

$$A + B \qquad \qquad \qquad [f, g]$$

↓

$$[f, g] \qquad \qquad \qquad C$$

↓

$$C \xleftarrow{\epsilon_C = [id_C, id_C]} C + C$$

↓

$$f \searrow \begin{matrix} (f \circ \iota_1) + (f \circ \iota_2) \\ \downarrow \\ A + B \end{matrix}$$

↓

$$(C, C) \qquad \qquad \qquad (f \circ \iota_1, f \circ \iota_2)$$

↓

$$(A, B)$$

$L : \mathcal{K}^I \rightarrow \mathcal{K}$  with  $L((A_i)_{i \in I}) = \coprod_{i \in I} A_i$  for all  $\mathcal{L}^I$ -objects and -morphisms  $(A_i)_{i \in I}$  is left adjoint to the diagonal functor  $\Delta_{\mathcal{K}}^I : \mathcal{K} \rightarrow \mathcal{K}^I$  (see [Functors](#)).

$$\begin{array}{ccc}
 (A_i)_{i \in I} & \xrightarrow{\eta_{(A_i)_{i \in I}} = (\iota_i)_{i \in I}} & (\coprod_{i \in I} A_i)_{i \in I} \\
 & \searrow & \downarrow \text{ } \Upsilon \\
 (f_i)_{i \in I} & & ([f_i]_{i \in I})_{i \in I} \\
 & & \downarrow \text{ } \Upsilon \\
 & & (B)_{i \in I} \\
 & & \downarrow \text{ } \Upsilon \\
 & & B
 \end{array}
 \qquad
 \begin{array}{ccc}
 & & \coprod_{i \in I} A_i \\
 & & \downarrow \text{ } \Upsilon \\
 & & [f_i]_{i \in I} \\
 & & \downarrow \text{ } \Upsilon \\
 & & B
 \end{array}$$
  

$$\begin{array}{ccc}
 B & \xleftarrow{\epsilon_B = [id_B]_{i \in I}} & B \times I \\
 & \swarrow & \downarrow \text{ } \lambda \\
 & f & \coprod_{i \in I} (f \circ \iota_i) \\
 & \searrow & \downarrow \text{ } \lambda \\
 & & \coprod_{i \in I} A_i
 \end{array}
 \qquad
 \begin{array}{ccc}
 & & (B)_{i \in I} \\
 & & \downarrow \text{ } \lambda \\
 & & (f \circ \iota_i)_{i \in I} \\
 & & \downarrow \text{ } \lambda \\
 & & (A_i)_{i \in I}
 \end{array}$$

## 11. Term functors (see chapter 9)

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature,  $U_S$  be the forgetful functor from  $Alg_\Sigma$  to  $Set^S$  (see chapter 7),  $V, V' \in Set^S$  and  $\mathcal{A}$  be a  $\Sigma$ -algebra.

$$\eta_V =_{def} inc_V : V \rightarrow T_\Sigma(V).$$

$$\frac{V \xrightarrow{g} U_S(\mathcal{A})}{T_\Sigma(V) \xrightarrow{g^*} \mathcal{A}}$$

$$\frac{T_\Sigma(V) \xrightarrow{h} \mathcal{A}}{V \xrightarrow{h^\# = h \circ \eta_V} U_S(\mathcal{A})}$$

By Theorem FREE, the functor

$$\begin{aligned} T_\Sigma : Set^S &\rightarrow Alg_\Sigma \\ V &\mapsto T_\Sigma(V) \\ f : V \rightarrow V' &\mapsto (\eta_{V'} \circ f)^* : T_\Sigma(V) \rightarrow T_\Sigma(V') \end{aligned}$$

is left adjoint to  $U_S$ . Proof of the functor property: Let  $g : V' \rightarrow V''$ . Then

$$(\eta_{V''} \circ g)^* \circ (\eta_{V'} \circ f)^* \circ \eta_V = (\eta_{V''} \circ g)^* \circ \eta_{V'} \circ f = \eta_{V''} \circ g \circ f.$$

Hence by Theorem FREE,

$$T_\Sigma(g \circ f) = (\eta_{V''} \circ g \circ f)^* = (\eta_{V''} \circ g)^* \circ (\eta_{V'} \circ f)^* = T_\Sigma(g) \circ T_\Sigma(f).$$

For all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$ , the co-unit  $\epsilon_{\mathcal{A}} = id_A^* : T_{\Sigma}(A) \rightarrow A$  folds each term  $t$  over  $A$  into an element of  $A$ , usually called the *value of  $t$  in  $\mathcal{A}$* .

Since  $V \in Set^S$  with  $V_s = \emptyset$  for all  $s \in S$  is initial in  $Set^S$  and left adjoints preserve initial objects,  $T_{\Sigma}$  is initial in  $Alg_{\Sigma}$ , which also follows from the definition of the extension

$$\textit{fold}^{\mathcal{A}} : T_{\Sigma} \rightarrow \mathcal{A}$$

(see  $\Sigma$ -terms and -coterm in chapter 9).

Since for all  $V \in Set^S$ , the (covariant) functors

$$F_V =_{def} Set^S(V, U_S(\_)) \quad \text{and} \quad Alg_{\Sigma}(T_{\Sigma}(V), \_)$$

are naturally equivalent,  $T_{\Sigma}(V)$  represents  $F_V$  (see chapter 5).

Let  $\Sigma(V)$  be defined as in chapter 9. There we have proved that  $T_{\Sigma}(V)$  is initial in  $Alg_{\Sigma(V)}$  and for all  $\Sigma(V)$ -algebras  $\mathcal{A}$  with carrier  $A$ ,  $\textit{fold}^{\mathcal{A}} = (\textit{val}^{\mathcal{A}})^*$ . Hence by the uniqueness of  $\textit{fold}^{\mathcal{A}}$ ,  $\textit{fold}^{\mathcal{A}} = id_A^* \circ T_{\Sigma}(\textit{val}^{\mathcal{A}})$ .

For instance, let  $\Sigma = \text{coStream}(X)$ . Then  $T_\Sigma \cong SF_X = \underline{\phantom{x}} \times X^*$  (see 19.3) and  $\Sigma(Y) = Dyn(X, Y)$ .  $SF_X(Y)$  is initial in  $\text{Alg}_{Dyn(X, Y)}$  and for all  $Dyn(X, Y)$ -algebras  $\mathcal{A}$  with carrier  $A$ ,  $\text{fold}^\mathcal{A} = (\alpha^\mathcal{A})^* = SF_X(\alpha^\mathcal{A}) = id_A^* \circ (\alpha^\mathcal{A} \times id_{X^*})$ .

Moreover,  $\Sigma(1) = List(X)$ ,  $X^*$  is initial in  $\text{Alg}_{List(X)}$  and for all  $List(X)$ -algebras  $\mathcal{A}$  with carrier  $A$ ,  $\text{fold}^\mathcal{A} = (\alpha^\mathcal{A})^* = SF_X(\alpha^\mathcal{A}) = id_A^* \circ (\alpha^\mathcal{A} \times id_{X^*}) = id_A^*$ .

## From term to state functors

Let  $\Sigma = (S, \mathcal{I}, D)$  be a comodel signature (see  $\Sigma$ -flowcharts),  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $V, V' \in Set^S$ . As an  $S$ -sorted function,

$$\text{return}_V^*: T_{\bar{\Sigma}}(V) \rightarrow (V \times A)^A$$

(see Term folding) defines a natural transformation from the term functor  $U_S T_{\bar{\Sigma}}$  to the state functor  $(\underline{\phantom{x}} \times A)^A$ :

$$\begin{array}{ccc}
 T_{\bar{\Sigma}}(V) & \xrightarrow{\text{return}_V^*} & (V \times A)^A \\
 \downarrow T_{\bar{\Sigma}}(f) & (1) & \downarrow (f \times A)^A \\
 T_{\bar{\Sigma}}(V') & \xrightarrow{\text{return}_{V'}^*} & (V' \times A)^A
 \end{array}$$

*Proof of (1) by structural induction.* Let  $f \in Set^S(V, V')$  and  $t \in T_{\bar{\Sigma}(V)}$  and  $a \in A_e$ .

*Case 1.*  $t \in V_s$  for some  $s \in S$ . Then for all  $a \in A_s$ .

$$\begin{aligned}
 & (f \times A)^A(return_V^*(t))(a) = (f \times A)^A(return_V(t))(a) \\
 &= (\lambda(x, a).(f(x), a) \circ return_V(t))(a) = (\lambda(x, a).(f(x), a))(return_V(t)(a)) \\
 &= (\lambda(x, a).(f(x), a))(t, a) = (f(t), a) = return_{V'}(f(t))(a) = return_{V'}^*(f(t))(a) \\
 &= return_{V'}^*((inc_{V'}(f(t)))(a) = return_{V'}^*((inc_{V'} \circ f)(t))(a) = return_{V'}^*((\eta_{V'} \circ f)(t))(a) \\
 &= return_{V'}^*((\eta_{V'} \circ f)^*(t))(a) = return_{V'}^*(T_{\bar{\Sigma}}(f)(t))(a).
 \end{aligned}$$

*Case 2.*  $t = \bar{d}(u)$  for some  $d : s \rightarrow s' \in D$  with  $s' \in S$  and  $u \in T_{\bar{\Sigma}(V)}(s')$ . Then

$$\begin{aligned}
 & (f \times A)^A(return_V^*(t)) = \lambda(x, a).(f(x), a) \circ return_V^*(t) \\
 &= \lambda(x, a).(f(x), a) \circ return_V^*(\bar{d}(u)) = \lambda(x, a).(f(x), a) \circ \bar{d}^{\mathcal{A}_V}(return_V^*(u)) \\
 &= \lambda(x, a).(f(x), a) \circ return_V^*(u) \circ d^{\mathcal{A}} = (f \times A)^A(return_V^*(u)) \circ d^{\mathcal{A}} \\
 &\stackrel{ind. \ hyp.}{=} return_{V'}^*(T_{\bar{\Sigma}}(f)(u)) \circ d^{\mathcal{A}} = return_{V'}^*((inc_{V'} \circ f)^*(u)) \circ d^{\mathcal{A}} \\
 &\stackrel{\text{Lemma } \textit{SUBST}}{=} (return_{V'}^* \circ inc_{V'} \circ f)^*(u) \circ d^{\mathcal{A}} = (return_{V'} \circ f)^*(u) \circ d^{\mathcal{A}} \\
 &= \bar{d}^{\mathcal{A}_{V'}}((return_{V'} \circ f)^*(u)) = (return_{V'} \circ f)^*(\bar{d}(u)) = (return_{V'} \circ f)^*(t)
 \end{aligned}$$

$$\begin{aligned}
&= (return_{V'}^* \circ inc_{V'} \circ f)^*(t) \stackrel{\text{Lemma } \text{SUBST}}{=} return_{V'}^*((inc_{V'} \circ f)^*(t)) \\
&= return_{V'}^*(T_{\bar{\Sigma}}(f)(t)).
\end{aligned}$$

Case 3.  $t = \bar{d}((i \rightarrow t_i \mid i \in I))$  for some  $d : s \rightarrow \coprod_{i \in I} s_i \in D$  and  $(t_i)_{i \in I} \in \bigtimes_{i \in I} T_{\bar{\Sigma}}(V)_{s_i}$ . Then

$$\begin{aligned}
&(f \times A)^A(return_V^*(t)) = \lambda(x, a).(f(x), a) \circ return_V^*(t) \\
&= \lambda(x, a).(f(x), a) \circ return_V^*(\bar{d}((i \rightarrow t_i \mid i \in I))) \\
&= \lambda(x, a).(f(x), a) \circ \bar{d}^{\mathcal{A}_V}(return_V^*((i \rightarrow t_i \mid i \in I))) \\
&= \lambda(x, a).(f(x), a) \circ [\pi_i(return_V^*((i \rightarrow t_i \mid i \in I))]_{i \in I} \circ d^{\mathcal{A}} \\
&= \lambda(x, a).(f(x), a) \circ [return_V^*(t_i)]_{i \in I} \circ d^{\mathcal{A}} = [\lambda(x, a).(f(x), a) \circ return_V^*(t_i)]_{i \in I} \circ d^{\mathcal{A}} \\
&= [(f \times A)^A(return_V^*(t_i))]_{i \in I} \circ d^{\mathcal{A}} \stackrel{\text{ind. hyp.}}{=} [return_{V'}^*(T_{\bar{\Sigma}}(f)(t_i))]_{i \in I} \circ d^{\mathcal{A}} \\
&= [return_{V'}^*((inc_{V'} \circ f)^*(t_i))]_{i \in I} \circ d^{\mathcal{A}} \stackrel{\text{Lemma } \text{SUBST}}{=} [(return_{V'}^* \circ inc_{V'} \circ f)^*(t_i)]_{i \in I} \circ d^{\mathcal{A}} \\
&= [(return_{V'} \circ f)^*(t_i)]_{i \in I} \circ d^{\mathcal{A}} = [\pi_i((return_{V'} \circ f)^*((i \rightarrow t_i \mid i \in I)))]_{i \in I} \circ d^{\mathcal{A}} \\
&= \bar{d}^{\mathcal{A}_{V'}}((return_{V'} \circ f)^*((i \rightarrow t_i \mid i \in I))) = (return_{V'} \circ f)^*(\bar{d}((i \rightarrow t_i \mid i \in I))) \\
&= (return_{V'} \circ f)^*(t) = (return_{V'}^* \circ inc_{V'} \circ f)^*(t) \\
&\stackrel{\text{Lemma } \text{SUBST}}{=} return_{V'}^*((inc_{V'} \circ f)^*(t)) = return_{V'}^*(T_{\bar{\Sigma}}(f)(t)). \quad \square
\end{aligned}$$

## 12. Varieties

Let  $\Sigma = (S, \mathcal{I}, F)$  be a **constructive** signature,  $R$  be a  $\Sigma$ -congruence on  $T_\Sigma(V)$  (or an isomorphic  $\Sigma$ -algebra; see [Products et al. of algebras](#)),  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $g \in A^V$  (see [Term folding](#)) such that for all  $(t, t') \in R$  and  $\sigma \in T_\Sigma(V)^V$ ,  $(\sigma^*(t), (\sigma^*(t')) \in R$ .

$g$  solves  $R$  if  $g$  solves all elements of  $R$ .

$\mathcal{A}$  satisfies  $R$ , written as  $\mathcal{A} \models R$ , if all  $g \in A^V$  solve  $R$ .

$T_\Sigma(V)/R$  satisfies  $R$ : Let  $g \in (T_\Sigma(V)/R)^V$ . Then  $nat_R \circ \sigma = g$  for some  $\sigma \in T_\Sigma(V)^V$ . Hence for all  $(t, t') \in R$ , [Lemma SUBST](#) implies

$$g^*(t) = (nat_R \circ \sigma)^*(t) = nat_R(\sigma^*(t')) = (nat_R \circ \sigma)^*(t') = g^*(t').$$

$Alg_{\Sigma, R}$ , the full subcategory of all  $Alg_\Sigma$  whose objects satisfy  $R$ , is called a **variety**.

Hence  $g$  solves  $R$  iff  $R \subseteq ker(g^*)$  iff  $g_R^* : T_\Sigma(V)/R \rightarrow \mathcal{A}$  is well-defined by

$$g_R^*(nat_R(t)) = g^*(t)$$

for all  $t \in T_\Sigma(V)$  iff  $g^* : T_\Sigma(V) \rightarrow \mathcal{A}$  factors through  $T_\Sigma(V)/R$ , i.e., there is  $g_R^*$  such that  $g^* = g_R^* \circ nat_R$ .

$$\begin{array}{ccccc}
 V & \xrightarrow{\text{inc}_V} & T_\Sigma(V) & \xrightarrow{\text{nat}_R} & T_\Sigma(V)/R \\
 & \searrow g & \downarrow g^* & \nearrow (2) & \nearrow g_R^* \\
 & & \mathcal{A} & &
 \end{array}$$

Since  $\text{nat}_\sim$  is epi, Lemma EMH (1) implies that  $g^*$  is  $\Sigma$ -homomorphic. Hence  $g_R^*$  is unique with (2), again because  $\text{nat}_R$  is epi.

We conclude that  $T_\Sigma(V)/R$  is **free in**  $\text{Alg}_{\Sigma,R}$ , i.e., for all  $\mathcal{A} \in \text{Alg}_{\Sigma,R}$  there is a unique  $\Sigma$ -homomorphism from  $T_\Sigma(V)/R$  to  $\mathcal{A}$  with (2).

In particular,  $T_\Sigma/R$  is initial in  $\text{Alg}_{\Sigma,R}$ .

Let  $\mathcal{B} = \mathcal{A}^{(A^V)}$ . By the last product equation,

$$\bigcap_{g \in A^V} \ker(g^*) = \ker(\langle g^* \rangle_{g \in A^V} : T_\Sigma(V) \rightarrow \mathcal{B}).$$

The  $\Sigma$ -algebra  $\mathcal{B}$  becomes a  $\Sigma(V)$ -algebra by defining  $\text{val}^\mathcal{B}(x)(g) = g(x)$  for all  $x \in V$  and  $g \in A^V$ . For the definition of  $\Sigma(V)$ , see [Term folding](#).

$$\begin{array}{ccccc}
 V & \xrightarrow{\quad inc_V \quad} & T_\Sigma(V) & \xrightarrow{\quad nat_R \quad} & T_\Sigma(V)/R \\
 g \downarrow & \nearrow (1) & \downarrow \langle g^* \rangle & \nearrow (4) & \dashrightarrow \\
 \mathcal{A} & \xleftarrow{\quad \pi_g \quad} & \mathcal{B} & \dashleftarrow & \langle g^* \rangle_R
 \end{array}$$

Hence the  $\Sigma$ -homomorphism  $\langle g^* \rangle$  is compatible with  $val$  and thus is also  $\Sigma(V)$ -homomorphic: For all  $x \in V$  and  $g' \in A^V$ ,

$$\begin{aligned}
 val^{\mathcal{B}}(\langle g^* \rangle(x))(g') &= val^{\mathcal{B}}(x)(g') = g'(x) \stackrel{(2)}{=} (g')^*(x) \stackrel{(3)}{=} \pi_{g'}(\langle g^* \rangle(x)) = \langle g^* \rangle(x)(g') \\
 &= \langle g^* \rangle(val^{T_\Sigma(V)}(x))(g').
 \end{aligned}$$

Therefore,  $fold^{\mathcal{B}} = \langle g^* \rangle$ .

Moreover,  $\mathcal{A} \models R$  iff for all  $g \in A^V$ , (2) holds true, iff  $R \subseteq \ker(\langle g^* \rangle)$  iff (4) holds true.

## Example

Let  $\Sigma = coStream(X)$  and  $Y$  be a set. Then

$$\Sigma(Y) = (\{state, X, Y\}, \{\delta : state \times X \rightarrow state, val : Y \rightarrow state\})$$

and thus  $\Sigma(Y)$  is equivalent to  $Dyn(X, Y)$ . Consequently,

$$T_\Sigma(Y) \cong Seq(X, Y) = (Y \times X^*, Op)$$

is initial in  $Alg_{Dyn}(X, Y)$  and for all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $Q$ ,  $fold^{\mathcal{A}^{(Q^Y)}} = \langle g^* \rangle_{g: Y \rightarrow Q}$  (see sample algebra 3, Example  $\prod Dyn$  in chapter 9 and adjunction 3 above).  $\square$

**Birkhoff Variety Theorem** (special case of [11], Theorem 6.2)

A class of  $\Sigma$ -algebras is a  $\Sigma$ -variety iff it is closed under the formation of subalgebras, homomorphic images and products.  $\square$

## Equational theories

Let  $E$  be an  $S$ -sorted set of  $\Sigma$ -equations (see Term folding).

For all  $s \in S$ ,  $E_s$  is supposed to consist of equations  $\varphi \Rightarrow t = t'$  with  $t, t' \in T_\Sigma(V)_s$ .

A  $(\Sigma, E)$ -algebra is a  $\Sigma$ -algebra that satisfies (all equations of)  $E$ .

$Alg_{\Sigma, E}$  denotes the full subcategory of  $Alg_\Sigma$  that consists of all  $(\Sigma, E)$ -algebras.

$RAlg_{\Sigma,E} =_{def} Alg_{\Sigma,E} \cap RAlg_{\Sigma}$  (see [Term folding](#)).

The least  $\Sigma$ -congruence  $R$  on  $T_{\Sigma}(V)$  such that for all  $\bigwedge_{i=1}^n t_i = t'_i \Rightarrow t = t' \in E$  and  $\sigma \in T_{\Sigma}(V)^V$ ,

$$\bigwedge_{i=1}^n (\sigma^*(t_i), \sigma^*(t'_i)) \in R \quad \text{implies} \quad (\sigma^*(t), \sigma^*(t')) \in R, \quad (5)$$

is called the **deductive theory of**  $(\Sigma, E)$  and denoted by  $DTh(E)$ .

The notion was introduced in [108] (for not only conditional equations, but also other Horn or Gentzen clauses; see chapter 10) to remind of the fact that  $DTh(E)$  captures the rules of equational deduction.

### Soundness of $DTh(E)$ w.r.t. $Alg_{\Sigma,E}$

For all  $e = (t, t') \in T_{\Sigma}(V)^2$ ,  $e \in DTh(E)$  implies  $Alg_{\Sigma,E} \models t = t'$ . (6)

*Proof.* By definition,  $DTh(E)$  coincides with the least fixpoint of

$$\begin{aligned} \Phi : \bigtimes_{s \in S} \mathcal{P}(T_{\Sigma}(V)_s^2) &\rightarrow \bigtimes_{s \in S} \mathcal{P}(T_{\Sigma}(V)_s^2) \\ R &\mapsto (inst(R_s) \cup cong(R_s) \cup \Delta_{T_{\Sigma}(V)_s}^2 \cup R_s^{-1} \cup R_s R_s)_{s \in S} \end{aligned}$$

where

$$\begin{aligned}
inst(R_s) &= \{(\sigma^*(t), \sigma^*(t')) \mid \bigwedge_{i=1}^n t_i = t'_i \Rightarrow t = t' \in E_s, \sigma \in T_\Sigma(V)^V, \\
&\quad \forall 1 \leq i \leq n : (\sigma^*(t_i), \sigma^*(t'_i)) \in R\}, \\
cong(R_s) &= \{(ft, ft') \mid f : e \rightarrow s \in F, (t, t') \in R_e\}.
\end{aligned}$$

Let  $\mathcal{A}$  be a  $(\Sigma, E)$ -algebra and  $R$  be the  $S$ -sorted set defined by  $R_s = \{(t, t') \in T_\Sigma(V)_s^2 \mid \mathcal{A} \models t = t'\}$  for all  $s \in S$ .

Since  $\text{lfp}(\Phi) = DTh(E)$ , fixpoint induction (see chapter 3) implies  $DTh(E) \subseteq R$  if  $R$  is  $\Phi$ -closed, i.e., if  $\Phi(R) \subseteq R$ .

So let  $s \in S$  and  $(t, t') \in \Phi(R_s)$ .

*Case 1:*  $(t, t') \in inst(R_s)$ . Then there are  $\bigwedge_{i=1}^n u_i = u'_i \Rightarrow u = u' \in E_s$  and  $\sigma \in T_\Sigma(V)^V$  such that  $t = \sigma^*(u)$ ,  $t' = \sigma^*(u')$  and  $(\sigma^*(u_i), \sigma^*(u'_i)) \in R$  for all  $1 \leq i \leq n$ . Hence Lemma **SUBST** implies

$$(g^* \circ \sigma)^*(u_i) = g^*(\sigma^*(u_i)) = g^*(\sigma^*(u'_i)) = (g^* \circ \sigma)^*(u'_i)$$

for all  $g \in A^V$ . Since  $\mathcal{A}$  satisfies  $E$ ,

$$g^*(t) = g^*(\sigma^*(u)) = (g^* \circ \sigma)^*(t) = (g^* \circ \sigma)^*(t') = g^*(\sigma^*(u')) = g^*(t'),$$

i.e.,  $(t, t') \in R_s$ .

*Case 2:*  $(t, t') \in \text{cong}(R_s)$ . Then  $t = fu$  and  $t' = fu'$  for some  $f : e \rightarrow s \in F$  and  $(u, u') \in R_e$ . Hence Lemma SUBST implies

$$g^*(t) = g^*(fu) = f^{\mathcal{A}}(g^*(u)) = f^{\mathcal{A}}(g^*(u')) = g^*(fu') = g^*(t'),$$

i.e.,  $(t, t') \in R_s$ .

*Case 3:*  $t = t'$ . Then  $(t, t') \in R_s$  because  $R_s$  is reflexive.

*Case 4:*  $(t', t) \in R_s$ . Then  $(t, t') \in R_s$  because  $R_s$  is symmetric.

*Case 5:*  $(t, u), (u, t') \in R_s$ . Then  $(t, t') \in R_s$  because  $R_s$  is transitive.

We conclude that  $R$  is  $\Phi$ -closed. □

$T_{\Sigma, E}(V) =_{\text{def}} T_{\Sigma}(V)/DTh(E)$  is a **free  $(\Sigma, E)$ -algebra over  $V$** , i.e., for all  $(\Sigma, E)$ -algebras  $\mathcal{A}$  with carrier  $A$  and  $g \in A^V$  there is a unique  $\Sigma$ -homomorphism  $g_E^* : T_{\Sigma, E}(V) \rightarrow \mathcal{A}$  with  $g_E^* \circ \text{nat}_{DTh(E)} = g^*$ .

$$\begin{array}{ccccc}
 V & \xrightarrow{\text{inc}_V} & T_\Sigma(V) & \xrightarrow{\text{nat}_{DTh(E)}} & T_{\Sigma,E}(V) \\
 & \searrow g & \downarrow g^* & \nearrow & \nearrow g_E^* \\
 & & \mathcal{A} & &
 \end{array}$$

(1)

*Proof.* Since  $DTh(E)$  is a  $\Sigma$ -congruence,  $T_{\Sigma,E}(V)$  is well-defined. Next we show that  $T_{\Sigma,E}(V)$  satisfies  $E$ . (7)

Let  $e = (\bigwedge_{i=1}^n t_i = t'_i \Rightarrow u = u') \in E$  and  $g \in T_{\Sigma,E}(V)^V$  such that  $g^*(t_i) = g^*(t'_i)$  for all  $1 \leq i \leq n$ . Then  $g = \text{nat} \circ \sigma$  for some  $\sigma : V \rightarrow T_\Sigma(V)$  and the natural map  $\text{nat} : T_\Sigma(V) \rightarrow T_{\Sigma,E}(V)$ . Since  $T_{\Sigma,E}(V)$  is a  $\Sigma$ -quotient of  $T_\Sigma(V)$ ,  $\text{nat}$  is  $\Sigma$ -homomorphic. Hence by Lemma SUBST,

$$\text{nat}(\sigma^*(t_i)) = (\text{nat} \circ \sigma)^*(t_i) = g^*(t_i) = g^*(t'_i) = (\text{nat} \circ \sigma)^*(t'_i) = \text{nat}(\sigma^*(t'_i)),$$

i.e.,  $(\sigma^*(t_i), \sigma^*(t'_i)) \in DTh(E)$ . Therefore,  $(\sigma^*(t), \sigma^*(t')) \in \text{inst}(DTh(E)) \subseteq \Phi(DTh(E))$  (see above).

Since  $DTh(E) = \text{lfp}(\Phi)$  is  $\Phi$ -closed,  $(\sigma^*(t), \sigma^*(t')) \in DTh(E)$ . Hence

$$g^*(t) = (\text{nat} \circ \sigma)^*(t) = \text{nat}(\sigma^*(t)) = \text{nat}(\sigma^*(t')) = (\text{nat} \circ \sigma)^*(t') = g^*(t').$$

We conclude that  $T_{\Sigma,E}(V)$  satisfies  $e$ .

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  that satisfies  $E$  and  $g \in A^V$ . Suppose that the kernel of  $g^* : T_\Sigma \rightarrow A$  is  $\Phi$ -closed.

Since  $DTh(E)$  is the least  $\Phi$ -closed subset of  $T_\Sigma(V)^2$ ,

$$DTh(E) \text{ is a subset of } \ker(g^*). \quad (8)$$

Hence  $g_E^* : T_{\Sigma,E} \rightarrow A$  is well-defined by  $g_E^* \circ \text{nat}_{DTh(E)} = g^*$ .

Since  $g^*$  is  $\Sigma$ -homomorphic and  $\text{nat}_{DTh(E)}$  is epi in  $Alg_\Sigma$ , Lemma EMH (2) implies that  $g_E^*$  is  $\Sigma$ -homomorphic. Let  $h : T_{\Sigma,E} \rightarrow A$  be any  $\Sigma$ -homomorphism with  $h \circ \text{nat}_{DTh(E)} = g^*$ . Hence  $h \circ \text{nat}_{DTh(E)} \circ inc_V = g^* \circ inc_V$ . Since  $g^*$  is the only  $\Sigma$ -homomorphism from  $T_\Sigma$  to  $\mathcal{A}$  with  $g^* \circ inc_V = g$ ,  $h \circ \text{nat}_{DTh(E)} = g_E^* \circ \text{nat}_{DTh(E)}$ . Since  $\text{nat}_{DTh(E)}$  is epi,  $h = g_E^*$ .

It remains to show that  $R = \ker(g^*)$  is  $\Phi$ -closed. So let  $s \in S$  and  $(t, t') \in \Phi(R_s)$ .

*Case 1:* There are  $\bigwedge_{i=1}^n u_i = u'_i \Rightarrow u = u' \in E_s$  and  $\sigma \in T_\Sigma(V)^V$  such that  $t = \sigma^*(u)$ ,  $t' = \sigma^*(u')$  and  $(\sigma^*(u_i), \sigma^*(u'_i)) \in R$  for all  $1 \leq i \leq n$ . Hence by Lemma SUBST,

$$g^*(t) = g^*(\sigma^*(u)) = (g^* \circ \sigma)^*(t) = (g^* \circ \sigma)^*(t') = g^*(\sigma^*(u')) = g^*(t'),$$

i.e.,  $(t, t') \in R_s$ .

*Case 2:*  $t = fu$  and  $t' = fu'$  for some  $f : e \rightarrow s \in F$  and  $(u, u') \in R_e$ . Hence by Lemma SUBST,

$$g^*(t) = g^*(fu) = f^A(g^*(u)) = f^A(g^*(u')) = g^*(fu') = g^*(t'),$$

i.e.,  $(t, t') \in R_s$ .

*Case 3:*  $t = t'$ . Then  $(t, t') \in R_s$  because  $R_s$  is reflexive.

*Case 4:*  $(t', t) \in R_s$ . Then  $(t, t') \in R_s$  because  $R_s$  is symmetric.

*Case 5:*  $(t, u), (u, t') \in R_s$ . Then  $(t, t') \in R_s$  because  $R_s$  is transitive.

We conclude that  $R$  is  $\Phi$ -closed. □

### Soundness and completeness of $DTh(E)$ w.r.t. $Alg_{\Sigma, E}$ and $T_{\Sigma, E}(V)$

For all  $e = (t, t') \in T_\Sigma(V)^2$ ,

$$e \in DTh(E) \text{ iff } Alg_{\Sigma, E} \models t = t' \text{ iff } T_{\Sigma, E}(V) \models t = t'.$$

*Proof.* Let  $e = (t, t') \in DTh(E)$  and  $\mathcal{A} \in Alg_{\Sigma, E}$ . By (6),  $\mathcal{A}$  satisfies  $t = t'$ .

Suppose that all  $(\Sigma, E)$ -algebras satisfy  $t = t'$ . By (7),  $T_{\Sigma, E}(V)$  satisfies  $E$ . Hence  $T_{\Sigma, E}(V)$  satisfies  $t = t'$ .

Suppose that  $T_{\Sigma,E}(V)$  satisfies  $t = t'$ . Let  $\text{nat}$  be the natural map from  $T_\Sigma(V)$  to  $T_{\Sigma,E}(V)$ . Then by Lemma SUBST,

$$\begin{aligned}\text{nat}(t) &= \text{nat}(\text{id}(t)) = \text{nat}(\text{inc}_V^*(t)) = (\text{nat} \circ \text{inc}_V)^*(t) = (\text{nat} \circ \text{inc}_V)^*(t') \\ &= \text{nat}(\text{inc}_V^*(t')) = \text{nat}(\text{id}(t')) = \text{nat}(t'),\end{aligned}$$

i.e.,  $(t, t') \in D\text{Th}(E)$ .

□

## Examples

**1.** Let  $\Sigma = \text{Mon}$ ,  $x, y, z \in V$  and

$$E = \{\text{mul}(x, \text{mul}(y, z)) = \text{mul}(\text{mul}(x, y), z), \text{mul}(x, \text{one}) = x, \text{mul}(\text{one}, x) = x\},$$

Then  $T_{\Sigma,E}(V) \cong V^*$ .

**2.** Let  $\Sigma = \text{List}(X)$ ,  $s \in V_{\text{list}}$  and

$$\begin{aligned}E &= \{\text{cons}(x, \text{cons}(y, s)) = \text{cons}(y, \text{cons}(x, s)) \mid x, y \in X\} \cup \\ &\quad \{\text{cons}(x, \text{cons}(x, s)) = \text{cons}(x, s) \mid x \in X\}.\end{aligned}$$

Then  $T_{\Sigma,E} \cong \mathcal{P}_\omega(X)$  is initial in  $\text{Alg}_{\Sigma,E}$  where  $\alpha^{\mathcal{P}_\omega(X)} = \emptyset$  and for all  $x \in X$  and finite subsets  $S$  of  $X$ ,  $\text{cons}^{\mathcal{P}_\omega(X)}(x, S) = S \cup \{x\}$ .

3. Let  $\Sigma = \text{ARRAY}$ ,  $s \in V_{list}$  and  $****$



$$ITh(E) =_{def} \{(t, t') \in T_\Sigma(V)^2 \mid \forall \sigma \in T_\Sigma^V : (\sigma^*(t), \sigma^*(t')) \in DTh(E) \cap T_\Sigma^2\}$$

is called the **inductive theory** of  $(\Sigma, E)$ , a notion introduced in [108, 109] to remind of the fact that the equations of  $ITh(E)$  can be proved by induction on  $T_\Sigma$ .

### Soundness and completeness of $ITh(E)$ w.r.t. $RAlg_{\Sigma, E}$ and $T_{\Sigma, E}$

For all  $e = (t, t') \in T_\Sigma(V)^2$ ,

$$e \in ITh(E) \text{ iff } RAlg_{\Sigma, E} \models t = t' \text{ iff } T_{\Sigma, E} \models t = t'.$$

*Proof.* Let  $e = (t, t') \in ITh(E)$ ,  $\mathcal{A}$  be a reachable  $\Sigma$ -algebra with carrier  $A$  and  $g \in A^V$ . Hence  $\sigma \in T_\Sigma^V$  is well-defined by  $g = fold^{\mathcal{A}} \circ \sigma$ , and thus  $(\sigma^*(t), \sigma^*(t')) \in DTh(E)$  and by (8),  $(\sigma^*(t), \sigma^*(t')) \in ker(fold^{\mathcal{A}})$ . Therefore, Lemma SUBST implies

$$g^*(t) = (fold^{\mathcal{A}} \circ \sigma)^*(t) = fold^{\mathcal{A}}(\sigma^*(t)) = fold^{\mathcal{A}}(\sigma^*(t')) = (fold^{\mathcal{A}} \circ \sigma)^*(t') = g^*(t').$$

Hence  $\mathcal{A}$  satisfies  $t = t'$ .

Suppose that all reachable  $(\Sigma, E)$ -algebras satisfy  $t = t'$ . By (7),  $T_{\Sigma, E}$  satisfies  $E$ . Since  $T_\Sigma$  is initial in  $Alg_\Sigma$  and  $\text{nat} = nat_{\sim_E} : T_\Sigma \rightarrow T_{\Sigma, E}$  is  $\Sigma$ -homomorphic,  $fold^{T_{\Sigma, E}} = nat$ . Hence  $T_{\Sigma, E}$  is reachable. We conclude that  $T_{\Sigma, E}$  satisfies  $t = t'$ .

Suppose that  $T_{\Sigma,E}$  satisfies  $t = t'$ . Let  $\sigma \in T_{\Sigma}^V$ . Hence by Lemma SUBST,

$$\begin{aligned} nat(\sigma^*(t)) &= fold^{T_{\Sigma,E}}(\sigma^*(t)) = (fold^{T_{\Sigma,E}} \circ \sigma)^*(t) = (fold^{T_{\Sigma,E}} \circ \sigma)^*(t') \\ &= fold^{T_{\Sigma,E}}(\sigma^*(t')) = nat(\sigma^*(t')), \end{aligned}$$

i.e.,  $(\sigma^*(t), \sigma^*(t')) \in DTh(E)$  and thus  $(t, t') \in ITh(E)$ .

□

### 13. Coterm functors (see chapter 9)

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature,  $U_S$  be the forgetful functor from  $Alg_\Sigma$  to  $Set^S$ ,  $C, C' \in Set^S$  and  $\mathcal{A}$  be a  $\Sigma$ -algebra.

$$\epsilon_C =_{def} \text{root} =_{def} \lambda t.t(\epsilon) : DT_\Sigma(C) \rightarrow C.$$

$$\frac{U_S(\mathcal{A}) \xrightarrow{g} C}{\mathcal{A} \xrightarrow{g^\#} DT_\Sigma(C)}$$

$$\frac{\mathcal{A} \xrightarrow{h} DT_\Sigma(C)}{U_S(\mathcal{A}) \xrightarrow{h^* = \epsilon_C \circ h} C}$$

By Theorem COFREE, the functor

$$DT_\Sigma : Set^S \rightarrow Alg_\Sigma$$

$$C \mapsto DT_\Sigma(C)$$

$$f : C' \rightarrow C \mapsto (f \circ \epsilon_{C'})^\# : DT_\Sigma(C') \rightarrow DT_\Sigma(C)$$

is right adjoint to  $U_S$ . Proof of the functor property: Let  $g : C'' \rightarrow C'$ . Then

$$\epsilon_C \circ (f \circ \epsilon_{C'})^\# \circ (g \circ \epsilon_{C''})^\# = f \circ \epsilon_{C'} \circ (g \circ \epsilon_{C''})^\# = f \circ g \circ \epsilon_{C''}.$$

Hence by Theorem COFREE,

$$DT_\Sigma(f \circ g) = (f \circ g \circ \epsilon_{C''})^\# = (f \circ \epsilon_{C'})^\# \circ (g \circ \epsilon_{C''})^\# = DT_\Sigma(f) \circ DT_\Sigma(g).$$

For all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $A$ , the unit  $\eta_A = id_A^\# : A \rightarrow DT_\Sigma(A)$  unfolds each element  $a$  of  $A$  into its “behavior tree” whose nodes are labelled by the “successors” of  $a$  w.r.t. the “transitions” induced by the interpretation in  $\mathcal{A}$  of the destructors of  $\Sigma$ .

Since  $V \in Set^S$  with  $V_s = 1$  for all  $s \in S$  is final in  $Set^S$  and right adjoints preserve final objects,  $DT_\Sigma$  is final in  $Alg_\Sigma$ , which also follows from the definition of the coextension

$$unfold^\mathcal{A} : \mathcal{A} \rightarrow DT_\Sigma$$

(see  $\Sigma$ -terms and -coterm in chapter 9).

Since for all  $V \in Set^S$ , the contravariant functors

$$F_C =_{def} Set^S(U_S(\_), C) \quad \text{and} \quad Alg_\Sigma(\_, DT_\Sigma(C))$$

are naturally equivalent,  $DT_\Sigma(C)$  represents  $F_C$  (see chapter 5).

Let  $\Sigma(C)$  be defined as in chapter 9. There we have proved that  $DT_\Sigma(C)$  is final in  $Alg_{\Sigma(C)}$  and for all  $\Sigma(C)$ -algebras  $\mathcal{A}$  with carrier  $A$ ,  $unfold^{\mathcal{A}} = (col^{\mathcal{A}})^\#$ . Hence by the uniqueness of  $unfold^{\mathcal{A}}$ ,  $unfold^{\mathcal{A}} = DT_\Sigma(col^{\mathcal{A}}) \circ id_A^\#$ .

For instance, let  $\Sigma = Med(X)$ . Then  $DT_\Sigma \cong BF_X = \underline{X}^*$  (see 19.4) and  $\Sigma(Y) = DAut(X, Y)$ .  $BF_X(Y)$  is final in  $Alg_{DAut(X, Y)}$  and for all  $DAut(X, Y)$ -algebras  $\mathcal{A}$  with carrier  $A$ ,  $unfold^{\mathcal{A}} = (\beta_{\mathcal{A}})^\# = BF_X(\beta_{\mathcal{A}}) = \beta_{\mathcal{A}}^{X^*} \circ id_A^\#$ .

Since  $\Sigma(2)$  is equivalent to  $Acc(X)$ ,  $Pow(X)$  is final in  $Alg_{Acc(X)}$  and for all  $Acc(X)$ -algebras  $\mathcal{A}$  with carrier  $A$ ,  $unfold^{\mathcal{A}} = \beta_{\mathcal{A}}^{X^*} \circ id_A^\# = \lambda a. \{w \in X^* \mid \beta^{\mathcal{A}}(id_A^\#(a)(w)) = 1\}$ .

## 14. Covarieties

Let  $\Sigma = (S, \mathcal{I}, F)$  be a **destructive** signature,  $I$  be a  $\Sigma$ -invariant of  $DT_\Sigma(C)$  (or an isomorphic  $\Sigma$ -algebra; see [Products et al. of algebras](#)),  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  and  $g \in C^A$  (see [State unfolding](#)) such that for all  $t \in I$  and  $\sigma \in C^{DT_\Sigma(C)}$ ,  $\sigma^\#(t) \in I$ .

$g$  solves  $I$  if for all  $a \in A$ ,  $g^\#(a) \in I$ .

$\mathcal{A}$  satisfies  $I$ , written as  $\mathcal{A} \models I$ , if all  $g \in C^A$  solve  $I$ .

$DT_\Sigma(C)|_I$  sattisfies  $I$ : Let  $g \in C^I$ . Then  $\sigma \circ inc_I = g$  for some  $\sigma \in C^{DT_\Sigma(C)}$ . Hence for all  $t \in I$ , [Lemma RECOL](#) implies  $g^\#(t) = (\sigma \circ inc_I)^\#(t) = \sigma^\#(t) \in I$ .

$Alg_{\Sigma, I}$ , the full subcategory of all  $Alg_\Sigma$  whose objects satisfy  $I$ , is called a **covariety**.

Hence  $g$  solves  $I$  iff  $img(g^\#) \subseteq I$  iff  $g_I^\# : \mathcal{A} \rightarrow I$  is well-defined by

$$g_I^\#(a) = g^\#(a)$$

for all  $a \in A$  iff  $g^\# : \mathcal{A} \rightarrow DT_\Sigma(C)$  factors through  $I$ , i.e., there is  $g_I^\#$  such that  $g^\# = inc_I \circ g_I^\#$ .

$$\begin{array}{ccccc}
 C & \xleftarrow{\text{root}} & DT_{\Sigma}(C) & \xleftarrow{\text{inc}_I} & DT_{\Sigma}(C)|_I \\
 & \nearrow g & \downarrow g^{\#} & \searrow (2) & \nearrow \tau \\
 & & \mathcal{A} & & \searrow g_I^{\#}
 \end{array}$$

Since  $\text{inc}_I$  is mono, Lemma EMH (2) implies that  $g^{\#}$  is  $\Sigma$ -homomorphic. Hence  $g_I^{\#}$  is unique with (2), again because  $\text{inc}_I$  is mono.

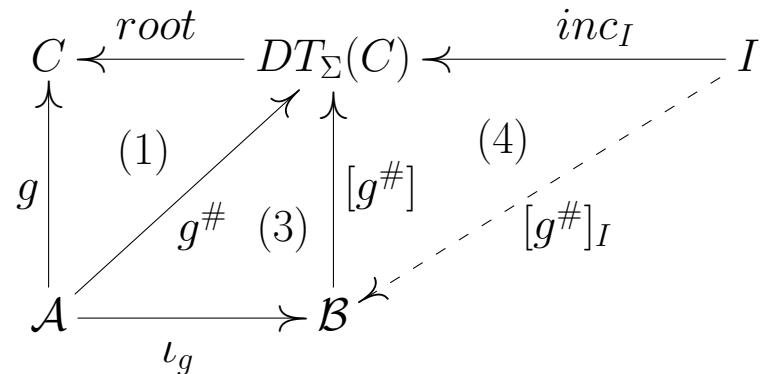
We conclude that  $DT_{\Sigma}(C)|_I$  is **cofree in  $\text{Alg}_{\Sigma,I}$** , i.e., for all  $\mathcal{A} \in \text{Alg}_{\Sigma,I}$  there is a unique  $\Sigma$ -homomorphism  $g_I^{\#} : \mathcal{A} \rightarrow DT_{\Sigma}(C)|_I$  with (2).

In particular,  $DT_{\Sigma}|_I$  is final in  $\text{Alg}_{\Sigma,I}$ .

Let  $\mathcal{B} = \mathcal{A} \times C^A$ . By the last sum equation,

$$\bigcup_{g \in C^A} \text{img}(g^{\#}) = \text{img}([g^{\#}]_{g \in C^A} : \mathcal{B} \rightarrow DT_{\Sigma}(C)).$$

The  $\Sigma$ -algebra  $\mathcal{B}$  becomes a  $\Sigma(C)$ -algebra by defining  $\text{col}^{\mathcal{B}}(a, g) = g(a)$  for all  $a \in A$  and  $g \in C^A$ . For the definition of  $\Sigma(C)$ , see State unfolding.



Hence the  $\Sigma$ -homomorphism  $[g^{\#}]$  is compatible with  $col$  and thus also  $\Sigma(C)$ -homomorphic:  
For all  $a \in A$  and  $g' \in C^A$ ,

$$\begin{aligned} [g^{\#}](col^B(a, g')) &= col^B(a, g') = g'(a) = root((g')^{\#}(a)) = root([g^{\#}](\iota_{g'}(a))) \\ &= col^{DT_{\Sigma}(C)}([g^{\#}](\iota_{g'}(a))) = col^{DT_{\Sigma}(C)}([g^{\#}](a, g')). \end{aligned}$$

Therefore,  $unfold^B = [g^{\#}]$ .

Moreover,  $\mathcal{A} \models I$  iff for all  $g \in C^A$ , (2) holds true, iff  $img([g^{\#}]) \subseteq I$  iff (4) holds true.

## Example

Let  $\Sigma = Med(X)$  and  $Y$  be a set. Then

$$\Sigma(Y) = (\{state, X, Y\}, \{\delta : Q \rightarrow Q^X, col : state \rightarrow Y\})$$

and thus  $\Sigma(Y)$  is equivalent to  $DAut(X, Y)$ . Consequently,

$$DT_\Sigma(Y) \cong Beh(X, Y) = (Y^{X^*}, Op)$$

is final in  $Alg_{DAut(X, Y)}$  and for all  $\Sigma$ -algebras  $\mathcal{A}$  with carrier  $Q$ ,  $unfold^{\mathcal{A} \times Y^Q} = [g^\#]_{g: Q \rightarrow Y}$  (see sample algebra 24, example  $\coprod DAut$  in chapter 9 and adjunction 4 above).  $\square$

**Birkhoff Covariety Theorem** (special case of [11], Theorem 6.15)

A class of  $\Sigma$ -algebras is a  $\Sigma$ -covariety iff it is closed under the formation of subalgebras, homomorphic images and coproducts.  $\square$

## Coequational theories

Let  $E$  be an  $S$ -sorted set of  $\Sigma$ -coequations (see State unfolding).

For all  $s \in S$ ,  $E_s$  is supposed to consist of coequations  $ex(t) \Rightarrow \varphi$  with  $t \in DT_\Sigma(C)_s$ .

A  $(\Sigma, E)$ -**algebra** is a  $\Sigma$ -algebra that satisfies (all coequations of)  $E$ .

$Alg_{\Sigma, E}$  denotes the full subcategory of  $Alg_\Sigma$  that consists of all  $(\Sigma, E)$ -algebras.

$OAlg_{\Sigma, E} =_{def} Alg_{\Sigma, E} \cap OAlg_{\Sigma}$  (see State unfolding).

The greatest  $\Sigma$ -invariant  $P$  of  $DT_{\Sigma}(C)$  such that for all  $ex(t) \Rightarrow \bigvee_{i=1}^n ex(t_i) \in E$  and  $\sigma \in C^{DT_{\Sigma}(C)}$ ,

$$\sigma^\#(t) \in P \quad \text{implies} \quad \bigvee_{i=1}^n \sigma^\#(t_i) \in P \quad (5)$$

is called the **deductive theory** of  $(\Sigma, E)$  and denoted by  $DTh(E)$ .

### Completeness of $DTh(E)$ w.r.t. $Alg_{\Sigma, E}$

For all  $t \in DT_{\Sigma}(C)$ ,  $Alg_{\Sigma, E} \not\models ex(t) \Rightarrow False$  implies  $t \in DTh(E)$ . (6)

*Proof.* By definition,  $DTh(E)$  coincides with the greatest fixpoint of

$$\begin{aligned} \Phi : \bigtimes_{s \in S} \mathcal{P}(DT_{\Sigma}(C)_s) &\rightarrow \bigtimes_{s \in S} \mathcal{P}(DT_{\Sigma}(C)_s) \\ P &\mapsto (coinst(P_s) \cap inv(P_s))_{s \in S} \end{aligned}$$

where

$$\begin{aligned} coinst(P_s) &= \{t \in DT_{\Sigma}(C)_s \mid \forall ex(u) \Rightarrow \bigvee_{i=1}^n ex(u_i) \in E_s, \sigma \in C^{DT_{\Sigma}(C)} : \\ &\quad \sigma^\#(t) = u \Rightarrow \exists 1 \leq i \leq n, t' \in P : \sigma^\#(t') = u_i\}, \end{aligned}$$

$$inv(P_s) = \{t \in DT_{\Sigma}(C)_s \mid \forall f : s \rightarrow e \in F : f^{DT_{\Sigma}(C)}(t) \in P_e\}.$$

Let  $P$  be the  $S$ -sorted set defined by  $P_s = \{t \in DT_{\Sigma}(C)_s \mid Alg_{\Sigma,E} \not\models ex(t) \Rightarrow False\}$  for all  $s \in S$ .

Since  $\text{gfp}(\Phi) = DTh(E)$ , fixpoint coinduction (see chapter 3) implies  $P \subseteq DTh(E)$  if  $P$  is  $\Phi$ -dense, i.e., if  $P \subseteq \Phi(P)$ .

So let  $s \in S$ ,  $t \in P_s$ ,  $ex(u) \Rightarrow \bigvee_{i=1}^n ex(u_i) \in E_s$  and  $\sigma \in C^{DT_{\Sigma}(C)}$  such that  $\sigma^{\#}(t) = u$ . Then  $Alg_{\Sigma,E} \not\models ex(t) \Rightarrow False$ , i.e.,  $g^{\#}(a) = t$  for some  $(\Sigma, E)$ -algebra  $\mathcal{A}$  with carrier  $A$ ,  $g \in C^A$  and  $a \in A_s$ . Hence by Lemma RECOL,

$$(\sigma \circ g^{\#})^{\#}(a) = \sigma^{\#}(g^{\#}(a)) = \sigma^{\#}(t) = u$$

and thus

$$\sigma^{\#}(g^{\#}(b)) = (\sigma \circ g^{\#})^{\#}(b) = u_i$$

for some  $b \in A$  and  $1 \leq i \leq n$  because  $\mathcal{A}$  satisfies  $E$ . Therefore,  $t' =_{def} g^{\#}(b) \in P$  and thus  $t \in \text{coinst}(P_s)$ .

Moreover, for all  $f : s \rightarrow e \in F$ ,

$$f^{DT_{\Sigma}(C)}(t) = f^{DT_{\Sigma}(C)}(g^{\#}(a)) = g^{\#}(f^{\mathcal{A}}(a)),$$

i.e.,  $f^{DT_{\Sigma}(C)}(t) \in P_e$ . Hence  $t \in \text{inv}(P_s)$ .

Therefore,  $t \in \Phi(P_s) = \text{coinst}(P_s) \cap \text{inv}(P_s)$ . We conclude that  $P$  is  $\Phi$ -dense. □

$DT_{\Sigma,E}(C) =_{def} DT_{\Sigma}(C)|_{DTh(E)}$  is a **cofree**  $(\Sigma, E)$ -**algebra over**  $C$ , i.e., for all  $(\Sigma, E)$ -algebras  $\mathcal{A}$  with carrier  $A$  and  $g \in C^A$  there is a unique  $\Sigma$ -homomorphism  $g_E^\# : \mathcal{A} \rightarrow DT_{\Sigma,E}(C)$  with  $inc_{DTh(E)} \circ g_E^\# = g^\#$ .

$$\begin{array}{ccccc}
 & & C & \xleftarrow{\text{root}} & DT_{\Sigma}(C) \\
 & \swarrow & \nearrow & & \nearrow inc_{DTh(E)} \\
 & g & (1) & g^\# & \\
 & & \uparrow & & \\
 & & \mathcal{A} & \dashleftarrow & \mathcal{T}
 \end{array}$$

In particular,  $DT_{\Sigma,E}$  is final in  $Alg\Sigma, E$ .

*Proof.* Since  $DTh(E)$  is a  $\Sigma$ -invariant,  $DT_{\Sigma,E}$  is well-defined. Next we show that  $DT_{\Sigma,E}(C)$  is a  $(\Sigma, E)$ -algebra. (7)

Let  $e = (ex(u) \Rightarrow \bigvee_{i=1}^n ex(u_i)) \in E$  and  $g \in C^{DTh(E)}$  such that  $g^\#(t) = u$  for some  $t \in DTh(E)$ . Then  $g = \sigma \circ inc$  for some  $\sigma \in C^{DT_{\Sigma}(C)}$  and the inclusion  $inc : DTh(E) \rightarrow DT_{\Sigma}(C)$ . Since  $DT_{\Sigma,E}(C)$  is a  $\Sigma$ -subalgebra of  $DT_{\Sigma}(C)$ ,  $inc$  is  $\Sigma$ -homomorphic. Hence by Lemma RECOL,

$$u = g^\#(t) = (\sigma \circ inc)^\#(t) = \sigma^\#(inc(t)) = \sigma^\#(t).$$

Since  $t \in DTh(E)$  and  $DTh(E) = gfp(\Phi)$  is  $\Phi$ -dense,  $t \in coinst(DTh(E))$ .

Therefore,  $\sigma^\#(t) = u$  implies  $\sigma^\#(t') = u_i$  for some  $1 \leq i \leq n$  and  $t' \in DTh(E)$ . Hence by Lemma RECOL,

$$g^\#(t') = (\sigma \circ inc)^\#(t') = \sigma^\#(inc(t')) = \sigma^\#(t') = u_i.$$

We conclude that  $DT_{\Sigma,E}(C)$  satisfies  $e$ .

Let  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$  that satisfies  $E$  and  $g \in C^A$ . Suppose that the image of  $g^\# : A \rightarrow DT_\Sigma$  is  $\Phi$ -dense.

Since  $DTh(E)$  is the greatest  $\Phi$ -dense subset of  $DT_\Sigma(C)$ ,

$$img(g^\#) \text{ is a subset of } DTh(E). \quad (8)$$

Hence  $\underline{g}_E^\# : A \rightarrow DTh(E)$  is well-defined by  $inc_{DTh(E)} \circ \underline{g}_E^\# = g^\#$ .

Since  $g^\#$  is  $\Sigma$ -homomorphic and  $inc_{DTh(E)}$  is mono in  $Alg_\Sigma$ , Lemma EMH (2) implies that  $\underline{g}_E^\#$  is  $\Sigma$ -homomorphic. Let  $h : A \rightarrow DTh(E)$  be any  $\Sigma$ -homomorphism with  $inc_{DTh(E)} \circ h = g^\#$ . Hence  $root \circ inc_{DTh(E)} \circ h = root \circ g^\# = g$ . Since  $g^\#$  is the only  $\Sigma$ -homomorphism from  $\mathcal{A}$  to  $DT_\Sigma(C)$  with  $root \circ g^\# = g$ ,  $inc_{DTh(E)} \circ h = inc_{DTh(E)} \circ \underline{g}_E^\#$ . Since  $inc_{DTh(E)}$  is mono,  $h = \underline{g}_E^\#$ .

It remains to show that  $\textcolor{red}{P} = \text{img}(g^\#)$  is  $\Phi$ -dense. So let  $s \in S$ ,  $t \in \textcolor{blue}{P}_s$ ,  $\text{ex}(u) \Rightarrow \bigvee_{i=1}^n \text{ex}(u_i) \in E_s$  and  $\sigma \in C^{DT_\Sigma(C)}$  such that  $\sigma^\#(t) = u$ . Then  $t = g^\#(a)$  for some  $a \in A$ . Hence by Lemma RECOL,

$$(\sigma \circ g^\#)^\#(a) = \sigma^\#(g^\#(a)) = \sigma^\#(t) = u$$

and thus

$$\sigma^\#(g^\#(b)) = (\sigma \circ g^\#)^\#(b) = u_i$$

for some  $b \in A$  and  $1 \leq i \leq n$  because  $\mathcal{A}$  satisfies  $E$ . Therefore,  $t' =_{\text{def}} g^\#(b) \in P$  and thus  $\textcolor{blue}{t} \in \text{coinst}(P_s)$ .

Moreover, for all  $f : s \rightarrow e \in F$ ,

$$f^{DT_\Sigma(C)}(t) = f^{DT_\Sigma(C)}(\text{unfold}^{\mathcal{A}}(a)) = \text{unfold}^{\mathcal{A}}(f^{\mathcal{A}}(a)),$$

i.e.,  $f^{DT_\Sigma(C)}(t) \in P_e$ . Hence  $\textcolor{blue}{t} \in \text{inv}(P_s)$ .

Therefore,  $t \in \Phi(P_s) = \text{coinst}(P_s) \cap \text{inv}(P_s)$ . We conclude that  $P$  is  $\Phi$ -dense.  $\square$

## Soundness and completeness of $DTh(E)$ w.r.t. $\text{Alg}_{\Sigma,E}$ and $DT_{\Sigma,E}(C)$

For all  $t \in DT_\Sigma(C)$ ,

$$\text{Alg}_{\Sigma,E} \not\models \text{ex}(t) \Rightarrow \text{False} \quad \text{iff} \quad t \in DTh(E) \quad \text{iff} \quad DT_{\Sigma,E}(C) \not\models \text{ex}(t) \Rightarrow \text{False}.$$

*Proof.* Suppose that  $Alg_{\Sigma, E}$  does not satisfy  $ex(t) \Rightarrow False$ . Then by (6),  $t \in DTh(E)$ . Let  $t \in DTh(E)$  and  $inc$  is the inclusion map from  $DTh(E)$  to  $DT_{\Sigma}(C)$ . Then by Lemma RECOL,  $(root \circ inc)^{\#}(t) = root^{\#}(inc(t)) = id(inc(t)) = t$ . Hence  $DT_{\Sigma, E}(C)$  does not satisfy  $ex(t) \Rightarrow False$ .

Suppose that  $DT_{\Sigma, E}(C)$  does not satisfy  $ex(t) \Rightarrow False$ . By (7),  $DT_{\Sigma, E}(C)$  satisfies  $E$ . Hence  $Alg_{\Sigma, E}$  does not satisfy  $ex(t) \Rightarrow False$ .  $\square$

## Examples

1. Let  $\Sigma = (\{state, 1\}, \{f : state \rightarrow state\})$  and  $\mathcal{A} = (A, \{\delta\})$  ??? be a  $\Sigma$ -algebra. Then for all  $a \in A$  and  $n \in \mathbb{N}$ ,

$$g^{\#}(a)(f^n) = (f^{DT_{\Sigma}(C)})^n(g^{\#}(a))(\epsilon) = g^{\#}(\delta^n(a))(\epsilon). \quad (9)$$

Let  $V_{state} = \{epsilon, x\}$ ,  $t = ()\{f \rightarrow t'\}$ ,  $t' = x\{f \rightarrow t'\}$ ,  $t_1 = ()\{f \rightarrow t\}$  and  $t_2 = x\{f \rightarrow t'\}$ .

**1.1** ([60], 6.1) Let  $\mathcal{K}$  be the category of  $\Sigma$ -algebras  $(A, \{\delta\})$  such that for all  $a \in A$  there is  $n > 0$  with  $\delta^n(a) = a$ .

$$\mathcal{K} = Alg_{\Sigma, \{ex(t) \Rightarrow False\}}.$$

*Proof.* Let  $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ . Then  $a = \delta^n(a)$  and  $g^\#(a) = t$  for some  $n > 0$ ,  $g \in V^A$  and  $a \in A$ . Hence

$$* = t(\epsilon) = g^\#(\delta^n(a))(\epsilon) \stackrel{(9)}{=} g^\#(a)(f^n) = t(f^n) = x. \not\models$$

Conversely, let  $\mathcal{A} \in Alg_{\Sigma} \setminus \mathcal{K}$ . Then  $a \neq \delta^n(a)$  for some  $a \in A$  and all  $n > 0$ . Let  $g = \lambda a'. if\ a' = a\ then\ * \ else\ x$ . Hence  $g^\#(a)(\epsilon) = g(a) = * = t(\epsilon)$  and for all  $n > 0$ ,

$$g^\#(a)(f^n) \stackrel{(9)}{=} g^\#(\delta^n(a))(\epsilon) = g(\delta^n(a)) = x = t(f) = t(f^n),$$

i.e.,  $g^\#(a) = t$ . Therefore,  $\mathcal{A} \notin Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

**1.2** ([60], 6.2) Let  $\mathcal{K}'$  be the category of  $\Sigma$ -algebras  $(A, \{\delta\})$  such that  $\delta$  is surjective.

$$\mathcal{K}' = Alg_{\Sigma, \{ex(t) \Rightarrow ex(t_1) \vee ex(t_2)\}}.$$

*Proof.* Let  $\mathcal{A} \in \mathcal{K}'$ . *Case 1:*  $\mathcal{A} \in \mathcal{K}$ . Then by 1.1,  $\mathcal{A}$  satisfies  $ex(t) \Rightarrow False$  and thus  $ex(t) \Rightarrow ex(t_1) \vee ex(t_2)$ .

*Case 2:*  $\mathcal{A} \notin \mathcal{K}$ . Then by 1.1,  $\mathcal{A}$  does not satisfy  $ex(t) \Rightarrow False$ , i.e.,  $g^\#(a) = t$  for some  $g \in V^A$  and  $a \in A$ . Since  $\delta$  is surjective, there is  $b \in A$  such that  $\delta(b) = a$ . Hence

$$g^\#(b) = g(b)\{f \rightarrow g^\#(\delta(b))\} = g(b)\{f \rightarrow g^\#(a)\} = g(b)\{f \rightarrow t\}$$

and thus  $g^\#(b) = t_1$  or  $g^\#(b) = t_2$  because  $g(b) \in \{epsilon, x\}$ .

We conclude that  $\mathcal{A}$  satisfies  $ex(t) \Rightarrow ex(t_1) \vee ex(t_2)$ .

Conversely, suppose that  $\mathcal{A}$  satisfies  $ex(t) \Rightarrow ex(t_1) \vee ex(t_2)$  and  $a \in A$ . *Case 1:*  $a = \delta^n(a)$  and  $g^\#(a) = t$  for some  $n > 0$ . Then  $a = \delta(b)$  for some  $b \in A$ .

*Case 2:* For all  $n > 0$ ,  $a \neq \delta^n(a)$ . Let  $g = \lambda a'. if\ a' = a\ then\ * else x$ . Then  $g^\#(a) = t$  (see the above proof of  $\mathcal{K} = Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ ). By assumption,  $g^\#(b) = t_1$  or  $g^\#(b) = t_2$  for some  $b \in A$ . Hence  $g^\#(\delta(b)) = f^{DT_\Sigma(C)}(g^\#(b)) = t$  and thus  $g(\delta(b)) = g^\#(\delta(b))(\epsilon) = t(\epsilon) = *$ . By the definition of  $g$ ,  $\delta(b) = a$ .

We conclude that  $\delta$  is surjective, i.e.,  $\mathcal{A} \in \mathcal{K}'$ .

**1.3** ([11], Ex. 4.15 (a)) Let  $t = ()\{f \rightarrow x\{f \rightarrow t\}\}$  and  $\mathcal{K}$  be the category of  $\Sigma$ -algebras  $(A, \{\delta\})$  such that for all  $a \in A$  there are  $k, n \in \mathbb{N}$  such that  $\delta^k(a) = \delta^{k+2n+1}(a)$ .

$\mathcal{K} = Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

*Proof.* Let  $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ . Then there are  $a \in A$  and  $k, n \in \mathbb{N}$  with  $g^\#(a) = t$  and  $\delta^{k+2n+1}(a) = \delta^k(a)$ . Hence

$$t(f^k) = g^\#(a)(f^k) \stackrel{(9)}{=} g^\#(\delta^k(a))(\epsilon) = g^\#(\delta^{k+2n+1}(a))(\epsilon) \stackrel{(9)}{=} g^\#(a)(f^{k+2n+1}) = t(f^{k+2n+1}). \checkmark$$

Conversely, let  $\mathcal{A} \in Alg_\Sigma \setminus \mathcal{K}$ . Then there is  $a \in A$  such that for all  $k, n \in \mathbb{N}$ , if  $\delta^k(a) = \delta^{k+n}(a)$ , then  $n$  is even. Let  $g = \lambda a'. if\ \exists n \in \mathbb{N} : a' = \delta^{2n}(a)\ then\ * else x$ .

$g$  is well-defined: Let  $k, n \in \mathbb{N}$  such that  $\delta^k(a) = \delta^{k+n}(a)$ . Then  $n$  is even. Hence  $k$  is even iff  $k + n$  is even, and thus  $g(\delta^k(a)) = g(\delta^{k+n}(a))$ . Moreover, for all  $n \in \mathbb{N}$ ,

$$g^\#(a)(f^{2n}) \stackrel{(9)}{=} g^\#(\delta^{2n}(a))(\epsilon) = g(\delta^{2n}(a)) = * = t(f^{2n}),$$

$$g^\#(a)(f^{2n+1}) \stackrel{(9)}{=} g^\#(\delta^{2n+1}(a))(\epsilon) = g(\delta^{2n+1}(a)) = x = t(f^{2n+1}),$$

i.e.,  $g^\#(a) = t$ . Therefore,  $\mathcal{A} \notin Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

**2.** ([7], 2.5; [11], Ex. 4.14; [151], Exs. 4.3 and 4.6) Let

$$\Sigma = (\{state, \{1, 2\}\}, \{f : state \rightarrow 1 + (state \times state)\}),$$

$V_{state} = \{epsilon, x\}$  and  $\mathcal{A} = (A, \{\delta\})$  ??? be a  $\Sigma$ -algebra.

**2.1** Let  $t = ()\{f \rightarrow 1(\epsilon)\}$  and  $\mathcal{K}$  be the category of  $\Sigma$ -algebras  $(A, \delta)$  such that for all  $a \in A$ ,  $\delta(a) = \iota_2(b, c)$  for some  $b, c \in A$ .

$\mathcal{K} = Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

*Proof.* Let  $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ . Then  $\delta(a) = \iota_2(b, c)$  and  $g^\#(a) = t$  for some  $b, c \in A$ ,  $g \in V^A$  and  $a \in A$ . Hence

$$() \{1 \rightarrow g^\#(b), 2 \rightarrow g^\#(c)\} = g^\#(b, c) = g^\#(\delta(a)) = f^{DT_\Sigma(C)}(g^\#(a)) = f^{DT_\Sigma(C)}(t) = 1(\epsilon). \sharp$$

Conversely, let  $\mathcal{A} \in \text{Alg}_\Sigma \setminus \mathcal{K}$  and  $g = \lambda a. \epsilon$ . Then  $\delta(a) = \epsilon$  for some  $a \in A$ . Hence

$$g^\#(a) = g(a)\{f \rightarrow 1(g^\#(\epsilon))\} = ()\{f \rightarrow 1(\epsilon)\} = t$$

and thus  $\mathcal{A} \notin \text{Alg}_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

**2.2** Let  $t = ()\{f \rightarrow 2(()\{1 \rightarrow t, 2 \rightarrow t\})\}$ ,  $t' = x\{f \rightarrow 2(()\{1 \rightarrow t, 2 \rightarrow t\})\}$ ,  $\mathcal{K}$  be the category of all  $\Sigma$ -algebras  $(A, \{\delta\})$  such that for all  $a \in A$ ,

$$\delta(b) = \epsilon \text{ for some } b \in \langle a \rangle, \quad (10)$$

and  $\mathcal{K}'$  be the category of all  $\Sigma$ -algebras  $(A, \{\delta\})$  such that for all  $a \in A$ , (10) holds true or  $\delta(a) = \iota_2(b)$  implies  $a \in \langle b \rangle$ .

$\mathcal{K} = \text{Alg}_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

*Proof.* Let  $\mathcal{A} \in \mathcal{K} \setminus \text{Alg}_{\Sigma, \{ex(t) \Rightarrow False\}}$ . Then  $g^\#(a) = t$  and (10) holds true for some  $g \in V^A$  and  $a \in A$ . Hence  $g^\#(b) = ()\{f \rightarrow 2(u)\}$  for some subcotermin  $u \neq *$  of  $t$ . Therefore,

$$* = g^\#(\epsilon) = g^\#(\epsilon) = g^\#(\delta(b)) = f^{DT_\Sigma(C)}(g^\#(b)) = f^{DT_\Sigma(C)}(()\{f \rightarrow 2(u)\}) = 2(u). \not\models$$

Conversely, let  $\mathcal{A} \in \text{Alg}_\Sigma \setminus \mathcal{K}$  and  $g = \lambda a. *$ . Then there is  $a \in A$  such that for all  $b \in \langle a \rangle$ ,  $\delta(b) \neq \epsilon$ . In particular,  $\delta(a) = \iota_2(b, c)$  for some  $b, c \in A$ .

Hence  $g^\#(a)(\epsilon) = g(a) = * = t(\epsilon)$ . Moreover, for all  $w \in \text{def}(g^\#(a))$  there is  $w' \in \{f, 1, 2\}^*$  such that  $w \in \{f * 1w', f * 2w', f * w'\}$ . If  $w = f * 1w'$ , then

$$g^\#(a)(w) = g^\#(b)(w') \stackrel{\text{ind. hyp.}}{=} t(w') = t(w).$$

If  $w = f * 2w'$ , then  $g^\#(a)(w) = g^\#(c)(w') \stackrel{\text{ind. hyp.}}{=} t(w') = t(w)$ . If  $w = f * w'$ , then both  $g^\#(a)(w)$  and  $t(w)$  are undefined. Hence  $g^\#(a) = t$  and thus  $\mathcal{A} \notin \text{Alg}_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

$$\mathcal{K}' = \text{Alg}_{\Sigma, \{ex(t') \Rightarrow False\}}.$$

*Proof.* Let  $\mathcal{A} \in \mathcal{K} \setminus \text{Alg}_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

**2.3** Let  $\mathcal{K}$  be the category of all  $\Sigma$ -algebras  $(A, \{\delta\})$  such that for all  $a, b, c \in A$ ,

$$\delta(a) = \iota_2(b, c) \Rightarrow \delta(b) \neq \epsilon \vee \delta(c) \neq \epsilon, \quad (11)$$

and  $t = ()\{f \rightarrow 2(()\{1 \rightarrow ()\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow ()\{f \rightarrow 1(\epsilon)\})\}$ .

$$\mathcal{K} = \text{Alg}_{\Sigma, \{ex(t) \Rightarrow False\}}.$$

*Proof.* Let  $\mathcal{A} \in \mathcal{K} \setminus \text{Alg}_{\Sigma, \{ex(t) \Rightarrow False\}}$ . Then  $g^\#(a) = t$  for some  $g \in V^A$  and  $a \in A$ . Hence

$$\begin{aligned} g^\#(\delta(a)) &= f^{DT_\Sigma(C)}(g^\#(a)) = f^{DT_\Sigma(C)}(t) = \lambda w.t(f * w) \\ &= ()\{1 \rightarrow ()\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow ()\{f \rightarrow 1(\epsilon)\}\} \end{aligned} \quad (12)$$

and thus  $\delta(a) = \iota_2(b, c)$  for some  $b, c \in A$ . Therefore,

$$\begin{aligned} ()\{1 \rightarrow g^\#(b), 2 \rightarrow g^\#(c)\} &= g^\#(b, c) = g^\#(\delta(a)) \\ &\stackrel{(12)}{=} ()\{1 \rightarrow ()\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow ()\{f \rightarrow 1(\epsilon)\}\}. \end{aligned} \tag{13}$$

Moreover, by (11) and w.l.o.g.,  $\delta(b) = \iota_2(d, e)$  for some  $d, e \in A$ .

By (13),  $g^\#(b) = ()\{f \rightarrow 1(\epsilon)\}$ . Hence

$$\begin{aligned} ()\{1 \rightarrow g^\#(d), 2 \rightarrow g^\#(e)\} &= g^\#(d, e) = g^\#(\delta(b)) = f^{DT_\Sigma(C)}(g^\#(b)) \\ &= \lambda w. g^\#(b)(f * w) = \lambda w. \text{if } w = \epsilon \text{ then } * \text{ else } (). \not\models \end{aligned}$$

Conversely, let  $A \in Alg_\Sigma \setminus \mathcal{K}$  and  $g = \lambda a.*$ . Then for some  $a \in A$ , (11) does not hold true, i.e., there are  $a, b, c \in A$  such that  $\delta(a) = \iota_2(b, c)$  and  $\delta(b) = \epsilon = \delta(c)$ . Hence

$$\begin{aligned} g^\#(a) &= g(a)\{f \rightarrow 2(()\{1 \rightarrow g^\#(b), 2 \rightarrow g^\#(c)\})\} \\ &= ()\{f \rightarrow 2(()\{1 \rightarrow g(b)\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow g(c)\{f \rightarrow 1(\epsilon)\}\})\} \\ &= ()\{f \rightarrow 2(()\{1 \rightarrow ()\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow ()\{f \rightarrow 1(\epsilon)\}\})\} = t \end{aligned}$$

and thus  $\mathcal{A} \notin Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

**2.4** Let  $\mathcal{K}$  be the category of all  $\Sigma$ -algebras  $A$  such that for all  $a, b, c \in A$ ,

$$\delta(a) = \iota_2(b, c) \wedge \delta(b) = \epsilon = \delta(c) \Rightarrow b = c, \tag{14}$$

and  $t = ()\{f \rightarrow 2(()\{1 \rightarrow ()\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow x\{f \rightarrow 1(\epsilon)\})\}\}$ .

$$\mathcal{K} = Alg_{\Sigma, \{ex(t) \Rightarrow False\}}.$$

*Proof.* Let  $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ . Then  $g^\#(a) = t$  for some  $g \in V^A$  and  $a \in A$ . Hence

$$\begin{aligned} g^\#(\delta(a)) &= f^{DT_\Sigma(C)}(g^\#(a)) = f^{DT_\Sigma(C)}(t) = \lambda w.t(f * w) \\ &= ()\{1 \rightarrow ()\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow x\{f \rightarrow 1(\epsilon)\}\} \end{aligned} \tag{15}$$

and thus  $\delta(a) = \iota_2(b, c)$  for some  $b, c \in A$ . Therefore,

$$\begin{aligned} ()\{\pi_1 \rightarrow g^\#(b), \pi_2 \rightarrow g^\#(c)\} &= g^\#(b, c) = g^\#(\delta(a)) \\ \stackrel{(15)}{=} ()\{1 \rightarrow ()\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow x\{f \rightarrow 1(\epsilon)\}\} \end{aligned} \tag{16}$$

and thus  $\delta(b) = \epsilon = \delta(c)$ . By (14),  $b = c$ . Hence

$$()\{f \rightarrow 1(\epsilon)\} \stackrel{(16)}{=} g^\#(b) = g^\#(c) \stackrel{(16)}{=} x\{f \rightarrow 1(\epsilon)\}. \not\models$$

Conversely, let  $A \in Alg_\Sigma \setminus \mathcal{K}$  and  $g = \lambda a'.if\ a' = c\ then\ x\ else\ *$ . Then for some  $a \in A$ , (14) does not hold true, i.e., there are  $a, b, c \in A$  such that  $\delta(a) = \iota_2(b, c)$ ,  $\delta(b) = \epsilon = \delta(c)$  and  $b \neq c$ . Hence

$$\begin{aligned}
g^\#(a) &= g(a)\{f \rightarrow 2(\{\} \{1 \rightarrow g^\#(b), 2 \rightarrow g^\#(c)\})\} \\
&= \{\} \{f \rightarrow 2(\{\} \{1 \rightarrow g(b)\{f \rightarrow 1(\epsilon)\}, 2 \rightarrow g(c)\{f \rightarrow 1(\epsilon)\}\})\} \\
&= \{\} \{f \rightarrow 2(\{\} \{1 \rightarrow (\{\} \{f \rightarrow 1(\epsilon)\}), 2 \rightarrow x\{f \rightarrow 1(\epsilon)\}\})\} = t
\end{aligned}$$

and thus  $\mathcal{A} \notin Alg_{\Sigma, \{ex(t) \Rightarrow False\}}$ .

**3.** (See [7], 2.6; sample final algebra 6) Let  $L \subseteq X^*$  and  $\mathcal{K}_L$  be the category of  $Acc(X)$ -algebras  $\mathcal{A}$  such that for all  $a \in A$ ,  $unfold^{\mathcal{A}}(a) \neq L$ .

**3.1** If  $L = X^*$ , then  $\mathcal{K}_L$  is the category of all  $Acc(X)$ -algebras  $\mathcal{A}$  with  $\beta^{\mathcal{A}} \neq \lambda a.0$ .

**3.2** If  $L = 1$ , then  $\mathcal{K}_L$  is the category of all  $Acc(X)$ -algebras  $\mathcal{A}$  with carrier  $A$  such that for all  $a \in A$  there is  $w \in def(id_A^\#(a))$  with  $w \neq \epsilon$  and  $\beta^{\mathcal{A}}(id_A^\#(a)(w)) = 1$ .

**4.** Let  $\Sigma = coList(X)$ ,  $t = \{\} \{split \rightarrow 1(x)\}$  for some  $x \in X$  and  $E = \{ex(t) \Rightarrow False\}$ . Then  $DT_{\Sigma, E} \cong X^{\mathbb{N}}$  is final in  $Alg_{\Sigma, E}$  where  $split^{X^{\mathbb{N}}}(f) = \iota_2(f(0), \lambda n.f(n + 1))$  for all  $f \in X^{\mathbb{N}}$ .

**5.** Let  $\Sigma = coList(X)$ ,  $t = \{\} \{split \rightarrow 2(\{\} \{1 \rightarrow x, 2 \rightarrow t\})\}$  for some  $x \in X$  and  $E = \{ex(t) \Rightarrow False\}$ . Then  $DT_{\Sigma, E} \cong X^*$  is final in  $Alg_{\Sigma, E}$  where  $split^{X^*}(\epsilon) = \epsilon$  and for all  $x \in X$  and  $w \in X^*$ ,  $split^{X^*}(x \cdot w) = \iota_2(x, split^{X^*}(w))$ . □

$$CTh(E) =_{def} \{\sigma^\#(t) \mid t \in DTh(E) \cap DT_\Sigma, \sigma \in V^{DT_\Sigma}\}$$

is called the **coinductive theory** of  $(\Sigma, E)$ .

**Soundness and completeness of  $CTh(E)$  w.r.t.  $OAlg_{\Sigma, E}$  and  $DT_{\Sigma, E}$**

For all  $t \in DT_\Sigma(C)$ ,

$$t \in CTh(E) \text{ iff } DT_{\Sigma, E} \not\models ex(t) \Rightarrow False \text{ iff } OAlg_{\Sigma, E} \not\models ex(t) \Rightarrow False.$$

*Proof.* Let  $t \in CTh(E)$ . Then  $\sigma^\#(u) = t$  for some  $t \in DTh(E) \cap DT_\Sigma$  and  $\sigma \in V^{DT_\Sigma}$ . Since  $DTh(E) \cap DT_\Sigma$  is the carrier of  $DT_{\Sigma, E}$ , we conclude that  $DT_{\Sigma, E}$  does not satisfy  $ex(t) \Rightarrow False$ .

Let  $DT_{\Sigma, E} \not\models ex(t) \Rightarrow False$ . Since  $unfold^{DT_{\Sigma, E}} = inc : DTh(E) \cap DT_\Sigma \rightarrow DT_\Sigma$ ,  $DT_{\Sigma, E}$  is observable. By (7),  $DT_{\Sigma, E}$  satisfies  $E$ . Hence  $OAlg_{\Sigma, E} \not\models ex(t) \Rightarrow False$ .

Suppose that some observable  $(\Sigma, E)$ -algebra  $\mathcal{A}$  with carrier  $A$  does not satisfy  $ex(t) \Rightarrow False$ . Then  $g^\#(a) = t$  for some  $a \in A$  and  $g \in V^A$ . Since  $\mathcal{A}$  is observable,  $\sigma \in V^{DT_\Sigma}$  is well-defined by  $g = \sigma \circ unfold^{\mathcal{A}}$ . Hence by Lemma RECOL,

$$\sigma^\#(unfold^{\mathcal{A}}(a)) = (\sigma \circ unfold^{\mathcal{A}})^\#(a) = g^\#(a) = t. \quad (17)$$

By (8),  $unfold^{\mathcal{A}}(a) \in DTh(E) \cap DT_\Sigma$ . Therefore, (17) implies  $t \in CTh(E)$ . □

## 15. Base algebra extensions

Let  $\Sigma = (S, \mathcal{I}, F)$  be a subsignature of a signature  $\Sigma' = (S', F')$  and  $B$  be a  $\Sigma$ -algebra.

For all  $e \in \mathcal{T}_p(S, \mathcal{I})$ ,  $e_B \in \mathcal{T}_p(S, \mathcal{I})$  is obtained from  $e$  by replacing each sort  $s \in S$  with  $B_s$ . Let  $F_B = \{f_B : e_B \rightarrow e'_B \mid f : e \rightarrow e' \in F'\}$ ,

$$\Sigma_B = (S' \setminus S, F_B).$$

Moreover,  $\sigma_B : \Sigma' \rightarrow \Sigma_B$  denotes the signature morphism that maps  $s \in S$  to  $B_s$ ,  $s \in S' \setminus S$  to  $s$  and  $f \in F'$  to  $f_B$ . Then for all  $\Sigma_B$ -algebras  $A$  and  $s \in S$ ,

$$(A|_{\sigma_B})_s = A_{\sigma_B(s)} = F_{\sigma_B(s)}(A) = \begin{cases} F_{B_s}(A) &= B_s \text{ if } s \in S, \\ F_s(A) &= A_s \text{ otherwise.} \end{cases}$$

Let  $U_\Sigma$  denote the **forgetful functor** from  $Alg_{\Sigma'}$  to  $Alg_\Sigma$ ,  $A$  be a  $\Sigma'$ -algebra and  $B = U_\Sigma(A)$ .  $A$  yields a  $\Sigma_B$ -algebra  $A_B$  that is defined as follows:

For all  $s \in S' \setminus S$ ,  $A_{B,s} = A_s$ , and for all  $f \in F'$ ,  $f_B^{A_{B,s}} = f^A$ .

The  $\sigma_B$ -reduct of  $A_B$  agrees with  $A$ :  $A_B|_{\sigma_B} = A$ .

Let  $\Sigma_B$  be constructive and  $\mu\Sigma$  be initial in  $Alg_{\Sigma_B}$ .

$U_\Sigma$  has a left adjoint  $L_{\Sigma'} : Alg_\Sigma \rightarrow Alg_{\Sigma'}$ :

For all  $\Sigma$ -algebras  $B$ ,  $L_{\Sigma'}(B) =_{def} \mu\Sigma|_{\sigma_B}$  is called the **free  $\Sigma'$ -algebra over  $B$** .

The unit  $\eta : Id \rightarrow U_\Sigma L_{\Sigma'}$  is defined as follows: For all  $b \in B$ ,  $\eta_B(b) = b$ .

The co-unit  $\epsilon : L_{\Sigma'} U_\Sigma \rightarrow Id$  is defined as follows: For all  $\Sigma$ -algebras  $B$  and  $\Sigma'$ -algebras  $A$ ,

$$L_{\Sigma'}(B) \xrightarrow{\epsilon_A} A = \mu\Sigma|_{\sigma_B} \xrightarrow{fold^{A_B}|_{\sigma_B}} A_B|_{\sigma_B}$$

where  $fold^{A_B}$  is the unique  $\Sigma_B$ -homomorphism from  $\mu\Sigma$  to  $A_B$ .

Let  $\Sigma_B$  be destructive and  $\nu\Sigma$  be final in  $Alg_{\Sigma_B}$ .

$U_\Sigma$  has a right adjoint  $R_{\Sigma'} : Alg_\Sigma \rightarrow Alg_{\Sigma'}$ :

For all  $\Sigma$ -algebras  $B$ ,  $R_{\Sigma'}(B) =_{def} \nu\Sigma|_{\sigma_B}$  is called the **cofree  $\Sigma'$ -algebra over  $B$** .

The co-unit  $\epsilon : U_\Sigma R_{\Sigma'} \rightarrow Id$  is defined as follows: For all  $b \in B$ ,  $\epsilon_B(b) = b$ .

The unit  $\eta : Id \rightarrow R_{\Sigma'} U_\Sigma$  is defined as follows: For all  $\Sigma$ -algebras  $B$  and  $\Sigma'$ -algebras  $A$ ,

$$A \xrightarrow{\eta_A} R_{\Sigma'}(B) = A_B|_{\sigma_B} \xrightarrow{unfold^{A_B}|_{\sigma_B}} \nu\Sigma|_{\sigma_B}$$

where  $unfold^{A_B}$  is the unique  $\Sigma_B$ -homomorphism from  $A_B$  to  $\nu\Sigma$ .

## Stream calculus

Let  $X$  be a semiring,  $C\Sigma = (\{list\}, \{X\}, C, \emptyset)$ ,  $\Sigma = (C\Sigma, Stream(X), \emptyset)$ ,

$$\begin{aligned}
 C &= \{\underline{=} : X \rightarrow list, \\
 &\quad \mathcal{X} : 1 \rightarrow list, \\
 &\quad \prec : X \times list \rightarrow list, \\
 &\quad +, *, \times, \otimes, \circ : list \times list \rightarrow list, \\
 &\quad \underline{-}^{-1} : list \rightarrow list, \\
 &\quad \underline{\underline{-}}^{-1} : list \rightarrow list, \\
 &\quad \sum_{n<\omega} : list^{\mathbb{N}} \rightarrow list, \\
 &\quad exp, sin, cos : list \rightarrow list\}
 \end{aligned}$$

and  $E$  be the following system of recursive equations: Let  $x, s, s' \in V$ .

$$\begin{array}{ll}
 head(\underline{x}) = x & tail(\underline{x}) = \underline{0} \\
 head(\mathcal{X}) = 0 & tail(\mathcal{X}) = \underline{1} \\
 head(x \prec s) = x & tail(x \prec s) = s \\
 head(s + s') = head(s) + head(s') & tail(s + s') = tail(s) + tail(s')
 \end{array}$$

$$\text{head}(s * s') = \text{head}(s) * \text{head}(s') \quad \text{tail}(s * s') = \text{tail}(s) * \text{tail}(s')$$

$$\text{head}(s \times s') = \text{head}(s) * \text{head}(s') \quad \text{tail}(s \times s') = (\text{tail}(s) \times s') + (\underline{\text{head}(s)} \times \text{tail}(s'))$$

convolution product

$$\text{head}(s \otimes s') = \text{head}(s) * \text{head}(s') \quad \text{tail}(s \otimes s') = (\text{tail}(s) \otimes s') + (s \otimes \text{tail}(s'))$$

shuffle product

$$\text{head}(s \circ s') = \text{head}(s) \quad \text{tail}(s \circ s') = \text{tail}(s') \times (\text{tail}(s) \circ s')$$

$$\text{head}(s^{-1}) = \text{head}(s)^{-1} \quad \text{tail}(s^{-1}) = (\underline{-1} \times \underline{\text{head}(s)^{-1}} \times \text{tail}(s)) \times s^{-1}$$

$$\text{head}(s^{\underline{-1}}) = \text{head}(s)^{-1} \quad \text{tail}(s^{\underline{-1}}) = \underline{-1} \times (\text{tail}(s) \otimes s^{-1} \otimes s^{-1})$$

$$\text{head}(\sum_{n < \omega} s_n) = \sum_{n < \omega} \text{head}(s_n) \quad \text{tail}(\sum_{n < \omega} s_n) = \sum_{n < \omega} \text{tail}(s_n)$$

$$\text{head}(\exp(s)) = \exp(\text{head}(s)) \quad \text{tail}(\exp(s)) = \text{tail}(s) \otimes \exp(s)$$

$$\text{head}(\sin(s)) = \sin(\text{head}(s)) \quad \text{tail}(\sin(s)) = \text{tail}(s) \otimes \cos(s)$$

$$\text{head}(\cos(s)) = \cos(\text{head}(s)) \quad \text{tail}(\cos(s)) = \text{tail}(s) \otimes -\sin(s)$$

Let  $a, b \in X^{\mathbb{N}}$ .  $-1$  and  $\text{head}(a)^{-1}$  are defined if  $X$  has unique additive and multiplicative inverses,  $a^{-1}$  is defined if  $a(0) \neq 0$  and  $a \circ b$  is defined if  $b(0) = 0$  (see [146], p. 14).  $\underline{-a} = \underline{-1} \times a$ .  $\exp(a)$ ,  $\sin(a)$ ,  $\cos(a)$  are defined if  $X \in \{\mathbb{R}, \mathbb{C}\}$ .

Given  $a_0, a_1, a_2, \dots \in X^{\mathbb{N}}$ ,  $\sum_{n < \omega} \text{head}(a_n)$  is defined only if  $X$  is a complete semiring or  $a_0, a_1, a_2, \dots$  is **summable**, i.e., for all  $i \in \mathbb{N}$ ,  $\sum_{n < \omega} a_n(i) = \lim_{n \rightarrow \infty} \sum_{k=0}^n a_k(i) \neq \infty$  (see [148], section 4).

For all  $x \in X$ ,  $a, b \in X^{\mathbb{N}}$  and  $n \in \mathbb{N}$ ,

$$\underline{x}(n) = \text{if } n = 0 \text{ then } x \text{ else } 0, \quad (1)$$

$$\mathcal{X}(n) = \text{if } n = 1 \text{ then } 1 \text{ else } 0, \quad (2)$$

$$(x \prec a)(n) = \text{if } n = 0 \text{ then } x \text{ else } a(n-1), \quad (3)$$

$$(a + b)(n) = a(n) + b(n), \quad (4)$$

$$(a \times b)(n) = \sum_{i=0}^n a(i) * b(n-i) \quad (5)$$

$$(a \otimes b)(n) = \sum_{i=0}^n \binom{n}{i} * a(i) * b(n-i) \quad (6)$$

*Proof.*

Since  $X^{\mathbb{N}}$  is final in  $\text{Alg}_{\text{Stream}}(X)$ , Theorem COINDSOL implies that, if the interpretation of  $\underline{x}, \mathcal{X}, \prec, +, \times, \otimes$  in  $X^{\mathbb{N}}$  given by (1)-(6) satisfies  $E$ , then (1)-(6) is the only solution of  $\overline{E}$  in  $X^{\mathbb{N}}$ . Indeed, (1)-(6) satisfies  $E$ :

$$\text{head}(\underline{x}) = \underline{x}(0) = x,$$

$$\text{tail}(\underline{x})(n) = \underline{x}(n+1) = 0 = \underline{0}(n),$$

$$\text{head}(\mathcal{X}) = \mathcal{X}(0) = 0,$$

$$\text{tail}(\mathcal{X})(n) = \mathcal{X}(n+1) = \text{if } n+1 = 1 \text{ then } 1 \text{ else } 0 = \text{if } n = 0 \text{ then } 1 \text{ else } 0 = \underline{1}(n),$$

$$\text{head}(x \prec a) = (x \prec a)(0) = x,$$

$$\text{tail}(x \prec a)(n) = (x \prec a)(n+1) = \text{if } n+1 = 0 \text{ then } x \text{ else } a(n)$$

$$= \text{if } n = -1 \text{ then } x \text{ else } a(n) = a(n)$$

$$\text{head}(a + b) = (a + b)(0) = a(0) + b(0) = \text{head}(a) + \text{head}(b),$$

$$\text{tail}(a + b)(n) = (a + b)(n+1) = a(n+1) + b(n+1) = \text{tail}(a)(n) + \text{tail}(b)(n)$$

$$= (\text{tail}(a) + \text{tail}(b))(n),$$

$$\text{head}(a \times b) = (a \times b)(0) = \sum_{i=0}^0 a(i) * b(0 - i) = a(0) * b(0) = \text{head}(a) * \text{head}(b),$$

$$\text{tail}(a \times b)(n) = (a \times b)(n + 1) = \sum_{i=0}^{n+1} a(i) * b(n + 1 - i)$$

$$= a(0) * b(n + 1) + \sum_{i=1}^{n+1} a(i) * b(n + 1 - i)$$

$$= a(0) * b(n + 1) + \sum_{i=0}^n a(i + 1) * b(n + 1 - (i + 1))$$

$$= a(0) * b(n + 1) + \sum_{i=0}^n a(i + 1) * b(n - i)$$

$$= \sum_{i=0}^n a(i + 1) * b(n - i) + a(0) * b(n + 1)$$

$$= \sum_{i=0}^n a(i + 1) * b(n - i) + a(0) * b(n + 1) + \sum_{i=1}^n 0 * b(n - i + 1)$$

$$= \sum_{i=0}^n a(i + 1) * b(n - i) + \underline{a(0)}(0) * b(n + 1) + \sum_{i=1}^n \underline{a(0)}(i) * b(n - i + 1)$$

$$= \sum_{i=0}^n a(i + 1) * b(n - i) + \sum_{i=0}^n \underline{a(0)}(i) * b(n - i + 1)$$

$$= \sum_{i=0}^n \text{tail}(a)(i) * b(n - i) + \sum_{i=0}^n \underline{a(0)}(i) * \text{tail}(b)(n - i)$$

$$= (\text{tail}(a) \times b)(n) + (\underline{a(0)} \times \text{tail}(b))(n)$$

$$= ((\text{tail}(a) \times b) + (\underline{\text{head}(a)} \times \text{tail}(b)))(n),$$

$$\text{head}(a \otimes b) = (a \otimes b)(0) = \sum_{i=0}^0 \binom{0}{i} * a(i) * b(0 - i) = 1 * a(0) * b(0)$$

$$= a(0) * b(0) = \text{head}(a) * \text{head}(b),$$

$$\begin{aligned}
tail(a \otimes b)(n) &= (a \times b)(n+1) = \sum_{i=0}^{n+1} \binom{n+1}{i} * a(i) * b(n+1-i) \\
&= \binom{n+1}{0} * a(0) * b(n+1) + \sum_{i=1}^{n+1} \binom{n+1}{i} * a(i) * b(n+1-i) \\
&= a(0) * b(n+1) + \sum_{i=0}^n \binom{n+1}{i+1} * a(i+1) * b(n-i) \\
&= a(0) * b(n+1) + \sum_{i=0}^n \binom{n}{i+1} * a(i+1) * b(n-i) + \sum_{i=0}^n \binom{n}{i} * a(i+1) * b(n-i) \\
&= a(0) * b(n+1) + \sum_{i=0}^n \binom{n}{i+1} * a(i+1) * b(n-(i+1)+1) \\
&\quad + \sum_{i=0}^n \binom{n}{i} * a(i+1) * b(n-i) \\
&= \sum_{i=0}^n \binom{n}{i} * a(i) * b(n-i+1) + \sum_{i=0}^n \binom{n}{i} * a(i+1) * b(n-i) \\
&= \sum_{i=0}^n \binom{n}{i} * a(i+1) * b(n-i) + \sum_{i=0}^n \binom{n}{i} * a(i) * b(n-i+1) \\
&= \sum_{i=0}^n \binom{n}{i} * tail(a)(i) * b(n-i) + \sum_{i=0}^n \binom{n}{i} * a(i) * tail(b)(n-i) \\
&= (tail(a) \otimes b)(n) + (a \otimes tail(b))(n) = ((tail(a) \otimes b) + (a \otimes tail(b)))(n).
\end{aligned}$$

□

For all  $x \in X$ ,  $a \in X^{\mathbb{N}}$  and summable  $\{a_n\}_{n < \omega} \subseteq X^{\mathbb{N}}$ ,

$$\underline{x} \times a = \lambda n.(x * a(n)) = \underline{x} \otimes a, \quad ([146], \text{ p. 24})$$

$$tail(\underline{x} \times a) = \underline{x} \times tail(a),$$

$$\mathcal{X} \times a = 0 \prec a, \quad ([130], \text{ equation (11)})$$

$$\underline{x} \times (y \times a) = \underline{x * y} \times a,$$

$$\underline{x} \times (y \prec a) = x * y \prec (\underline{x} \times a),$$

$$\sum_{n < \omega} a_n = a_0 + \sum_{n < \omega} a_{n+1}, \quad (\text{proof by coinduction})$$

$$\sum_{n < \omega} a \times a_n = a \times \sum_{n < \omega} a_n. \quad (\text{proof by coinduction})$$

For all  $n \in \mathbb{N}$ , define  $a^0 = a^{\underline{0}} = \underline{1}$ ,  $a^{\textcolor{red}{n+1}} = a \times a^n$  and  $a^{\textcolor{red}{n+1}} = a \otimes a^{\underline{n}}$ . By coinduction,

$$a(0) = 0 \Rightarrow tail(a^{n+1}) = tail(a) \times a^n, \quad (7)$$

$$tail(a^{n+1}) = \underline{n+1} \otimes tail(a) \otimes a^{\underline{n}}. \quad (8)$$

By (2) and since for all  $i, k \in \mathbb{N}$ ,  $\mathcal{X}(i) * \mathcal{X}^n(k - i) \neq 0 \Leftrightarrow i = 1 \wedge k = n + 1$ , a proof by induction on  $n$  yields

$$\mathcal{X}^n = \lambda i. (if\ i = n\ then\ 1\ else\ 0), \quad \mathcal{X}^{\underline{n}} = \underline{n!} \times \mathcal{X}^n$$

([146], Equation (30)) and thus for all  $x \in X$ :

$$\underline{x} \times \mathcal{X}^n = \lambda i. (if\ i = n\ then\ x\ else\ 0).$$

## Representations of $X^{\mathbb{N}}$

Let  $X$  be a group. Then  $B = X^{\mathbb{N}}$  is the carrier of a  $Stream(X)$ -algebra:  $head^B(a) = a(0)$  and  $tail^B(a) = \lambda n. (a(n+1) - a(n))$ .

The function  $\tau : \langle head^B, tail^B \rangle \rightarrow X^{\mathbb{N}}$  that maps  $a \in X^{\mathbb{N}}$  to the stream  $\lambda n. \sum_{i=0}^n \binom{-n}{i} * a(i)$  is a  $Stream(X)$ -isomorphism ([130], section 1.2).

The inverse of  $\tau$  maps  $a \in X^{\mathbb{N}}$  to  $\lambda n. \sum_{i=0}^n \binom{n}{i} * a(i)$ .

The set  $\mathbb{A}$  of functions  $f : \mathbb{R} \rightarrow \mathbb{R}$  that are analytic at 0 provides the carrier of a  $Stream(X)$ -algebra:  $head^{\mathbb{A}}(f) = f(0)$  and  $tail^{\mathbb{A}}(f) = Df$  (first derivative of  $f$ ).

The **Taylor transform**  $T : \mathbb{A} \rightarrow \mathbb{R}^{\mathbb{N}}$  that maps  $f \in \mathbb{A}$  to the stream  $\lambda n.(D^n f)(0)$  is a  $Stream(X)$ -monomorphism ([130], section 2.2; [146], section 5; [147], section 3.4; [148], section 12). The inverse  $T^{-1}$  of  $T$  is defined on the image of  $T$  and maps  $a \in T(\mathbb{A})$  to the power series  $\lambda x. \sum_{i < \omega} a(i)x^i/i!$ . The streams of  $T(\mathbb{A})$  are called streams of **Taylor coefficients**.

**Sample analytic functions** Let // denote integer division.

$$\exp = \lambda x. \sum_{n < \omega} a(n) * x^n \text{ where } a(n) = 1/n!$$

$$\sin = \lambda x. \sum_{n < \omega} a(n) * x^n \text{ where } a(n) = \text{if even}(n) \text{ then } 0 \text{ else } (-1)^{n//2}/n!$$

$$\cos = \lambda x. \sum_{n < \omega} a(n) * x^n \text{ where } a(n) = \text{if odd}(n) \text{ then } 0 \text{ else } (-1)^{n//2}/n!$$

Proofs by coinduction yield for all  $f, g \in \mathbb{A}$ :

$$T(f + g) = T(f) + T(g) \quad (9)$$

$$T(f * g) = T(f) \otimes T(g) \quad (10)$$

$$T(\lambda x. \int_0^x f) = \mathcal{X} \times T(f) \quad (11)$$

The function  $g : \mathbb{R}^{\mathbb{N}} \rightarrow \mathbb{R}^{\mathbb{N}}$  that maps  $a \in \mathbb{R}^{\mathbb{N}}$  to the stream  $\lambda n.(n! * a(n))$  is a bijection that is compatible with the stream operators  $=_{\underline{=}}$ ,  $\mathcal{X}$ ,  $\prec$  and  $+$  and satisfies

$$g(a \times b) = g(a) \otimes g(b) \quad \text{and} \quad g(a^{-1}) = g(a)^{-1}$$

([130], section 3.1; [146], Thm. 6.4 where  $g = \Delta_{\mathbb{R}^{\mathbb{N}}}$ ; [148], Thm. 10.1 where  $g = \Lambda_c$ ). The inverse of  $g$  maps  $a \in \mathbb{R}^{\mathbb{N}}$  to the stream  $\lambda n.(a(n)/n!)$ .

Hence the composition of  $T^{-1} \circ g$  maps a stream  $a \in \mathbb{R}^{\mathbb{N}}$  of Taylor coefficients to its **generating function**  $\lambda x. \sum_{n < \omega} a(n) * x^n$  ([130], section 3.1). The inverse  $g^{-1} \circ T$  sends  $f \in \mathbb{A}$  to  $\lambda n.(D^n f / n!)$ .

## Fundamental Theorem of $\text{Stream}(X)$

Let  $A$  be the final  $\text{Stream}(X)$ -algebra (with carrier  $X^{\mathbb{N}}$ ).  $A$  satisfies the following set  $E$  of equations: Let  $s, s' \in V$  and for all  $n \in \mathbb{N}$ ,  $s(n) = \text{head}(\text{tail}^n(s))$ .

$$s = \underline{\text{head}(s)} + (\mathcal{X} \times \text{tail}(s)), \quad (12)$$

$$s = \sum_{n < \omega} \underline{s(n)} \times \mathcal{X}^n, \quad (13)$$

$$s = \sum_{n < \omega} \underline{s(n)/n!} \times \mathcal{X}^n, \quad (14)$$

$$\text{head}(s') = 0 \Rightarrow s \circ s' = \sum_{n < \omega} \underline{s(n)} \times s'^n, \quad (15)$$

$$\exp(s) = \sum_{n < \omega} \underline{1/n!} \times s^n, \quad (16)$$

$$\exp(s + s') = \exp(s) \otimes \exp(s'), \quad (17)$$

$$\sin(s) = \sum_{n < \omega} \underline{(-1)^n / (2n+1)!} \times s^{2n+1}, \quad (18)$$

$$\cos(s) = \sum_{n < \omega} \underline{(-1)^n / (2n)!} \times s^{2n}, \quad (19)$$

$$\sin(s)^2 + \cos(s)^2 = \underline{1}. \quad (20)$$

*Proof.*

We show that  $\sim = \{(t^A(a), u^A(a)) \mid t = u \in E, a \in A_{src(t)}\}$  is a  $Stream(X)$ -bisimulation modulo  $C$ . Hence by coinduction on  $\sim$ ,  $A$  satisfies  $E$ .

Besides the equations given above, we also use some of those listed in [147], Thm. 2.4.1 or 3.1.1, and involving  $+$ ,  $\times$ ,  $\otimes$ ,  $\underline{0}$  or  $\underline{1}$  and Let  $a, b \in X^{\mathbb{N}}$ . We identify the arrows of  $\Sigma$  with their interpretations in  $A$ .

(12)

$$\begin{aligned} & head(\underline{head}(a) + (\mathcal{X} \times tail(a))) = head(\underline{a(0)}) + head(\mathcal{X} \times tail(a)) \\ &= head(\underline{a(0)}) + head(0 \prec tail(a)) = head(\underline{a(0)}) + 0 = head(\underline{a(0)}) = a(0) = head(a), \\ & tail(\underline{head}(a) + (\mathcal{X} \times tail(a))) = tail(\underline{a(0)}) + tail(\mathcal{X} \times tail(a)) = \underline{0} + tail(\mathcal{X} \times tail(a)) \\ &= tail(\mathcal{X} \times tail(a)) = tail(0 \prec tail(a)) = tail(a) \end{aligned}$$

(see [146], Thm. 5.1; [148], Thm. 4.1).

(13)

$$\begin{aligned}
& \text{head}(\sum_{n<\omega} \underline{a(n)} \times \mathcal{X}^n) = \sum_{n<\omega} \text{head}(\underline{a(n)} \times \mathcal{X}^n) = \sum_{n<\omega} \text{head}(\underline{a(n)}) * \text{head}(\mathcal{X}^n) \\
&= \sum_{n<\omega} a(n) * \text{head}(\mathcal{X}^n) = \sum_{n<\omega} a(n) * \mathcal{X}^n(0) = a(0) = \text{head}(a), \\
& \text{tail}(\sum_{n<\omega} \underline{a(n)} \times \mathcal{X}^n) = \sum_{n<\omega} \text{tail}(\underline{a(n)} \times \mathcal{X}^n) = \sum_{n<\omega} \underline{a(n)} \times \text{tail}(\mathcal{X}^n) \\
&= (\underline{a(0)} \times \text{tail}(\mathcal{X}^0)) + \sum_{n<\omega} \underline{a(n+1)} \times \text{tail}(\mathcal{X}^{n+1}) \\
&= (\underline{a(0)} \times \underline{0}) + \sum_{n<\omega} \underline{a(n+1)} \times \mathcal{X}^n = \sum_{n<\omega} \underline{a(n+1)} \times \mathcal{X}^n \\
&= \sum_{n<\omega} \underline{\text{tail}(a)(n)} \times \mathcal{X}^n \sim \text{tail}(a)
\end{aligned}$$

(see [146], Thm. 5.2; [148], Thm. 4.3).

(14) By (13),

$$\begin{aligned}
a &= \sum_{n<\omega} \underline{a(n)} \times \mathcal{X}^n = \sum_{n<\omega} \underline{n! * a(n)/n!} \times \mathcal{X}^n = \sum_{n<\omega} \underline{a(n)/n!} \times (\underline{n!} \times \mathcal{X}^n) \\
&= \sum_{n<\omega} \underline{a(n)/n!} \times \mathcal{X}^n.
\end{aligned}$$

(15)

$$\begin{aligned} \text{head}(\sum_{n<\omega} \underline{a(n)} \times b^n) &= \sum_{n<\omega} \text{head}(\underline{a(n)} \times b^n) = \sum_{n<\omega} \text{head}(\underline{a(n)}) * \text{head}(b^n) \\ &= \sum_{n<\omega} a(n) * b^n(0) = \sum_{n<\omega} a(n) * b(0)^n \stackrel{b(0)=0}{=} a(0) * 1 = a(0) = \text{head}(a) = \text{head}(a \circ b), \end{aligned}$$

$$\begin{aligned} \text{tail}(\sum_{n<\omega} \underline{a(n)} \times b^n) &= \sum_{n<\omega} \text{tail}(\underline{a(n)} \times b^n) = \sum_{n<\omega} \underline{a(n)} \times \text{tail}(b^n) \\ &= (\underline{a(0)} \times \text{tail}(b^0)) + \sum_{n<\omega} \underline{a(n+1)} \times \text{tail}(b^{n+1}) \\ &= (\underline{a(0)} \times \underline{0}) + \sum_{n<\omega} \underline{a(n+1)} \times \text{tail}(b^{n+1}) = \sum_{n<\omega} \underline{a(n+1)} \times \text{tail}(b^{n+1}) \\ &\stackrel{(7)}{=} \sum_{n<\omega} \underline{\text{tail}(a)(n)} \times (\text{tail}(b) \times b^n) = \text{tail}(b) \times \sum_{n<\omega} \underline{\text{tail}(a)(n)} \times b^n \\ &\sim_C \text{tail}(b) \times (\text{tail}(a) \circ b) = \text{tail}(a \circ b) \end{aligned}$$

(see [147], Thm. 2.5.3).

(16)

$$\begin{aligned}
& \text{head}(\sum_{n<\omega} \underline{1/n!} \times \underline{a^n}) = \sum_{n<\omega} \text{head}(\underline{1/n!} \times \underline{a^n}) = \sum_{n<\omega} \text{head}(\underline{1/n!}) * \text{head}(\underline{a^n}) \\
&= \sum_{n<\omega} (1/n!) * a^n(0) = \sum_{n<\omega} (1/n!) * a(0)^n = \exp(a(0)) = \text{head}(\exp(a)), \\
& \text{tail}(\sum_{n<\omega} \underline{1/n!} \times \underline{a^n}) = \sum_{n<\omega} \text{tail}(\underline{1/n!} \times \underline{a^n}) = \sum_{n<\omega} \underline{1/n!} \times \text{tail}(\underline{a^n}) \\
&= \underline{1/0!} \times \text{tail}(a^0) + \sum_{n<\omega} \underline{1/(n+1)!} \times \text{tail}(a^{n+1}) \\
&= \underline{1} \times \underline{0} + \sum_{n<\omega} \underline{1/(n+1)!} \times \text{tail}(a^{n+1}) = \sum_{n<\omega} \underline{1/(n+1)!} \times \text{tail}(a^{n+1}) \\
&\stackrel{(8)}{=} \sum_{n<\omega} \underline{1/(n+1)!} \times (\underline{n+1} \otimes \text{tail}(a) \otimes \underline{a^n}) \\
&= \sum_{n<\omega} \underline{1/(n+1)!} \otimes \underline{n+1} \otimes \text{tail}(a) \otimes \underline{a^n} \\
&= \text{tail}(a) \otimes \sum_{n<\omega} \underline{1/(n+1)!} \otimes \underline{n+1} \otimes \underline{a^n} \\
&= \text{tail}(a) \otimes \sum_{n<\omega} \underline{(n+1)/(n+1)!} \otimes \underline{a^n} = \text{tail}(a) \otimes \sum_{n<\omega} \underline{1/n!} \otimes \underline{a^n} \\
&\sim_C \text{tail}(a) \otimes \exp(a) = \text{tail}(\exp(a)).
\end{aligned}$$

(19)

$$\begin{aligned}
& \text{head}(\exp(a + b)) = \exp(\text{head}(a + b)) = \exp(\text{head}(a) + \text{head}(b)) \\
&= \exp(\text{head}(a)) * \exp(\text{head}(b)) = \text{head}(\exp(a)) * \text{head}(\exp(b)) \\
&= \text{head}(\exp(a) \otimes \exp(b)),
\end{aligned}$$

$$\begin{aligned}
& \text{tail}(\exp(a + b)) = \text{tail}(a + b) \otimes \exp(a + b) = (\text{tail}(a) + \text{tail}(b)) \otimes \exp(a + b) \\
&= (\text{tail}(a) \otimes \exp(a + b)) + (\text{tail}(b) \otimes \exp(a + b)) \\
&\sim_C (\text{tail}(a) \otimes \exp(a) \otimes \exp(b)) + (\text{tail}(b) \otimes \exp(a) \otimes \exp(b)) \\
&= (\text{tail}(a) \otimes \exp(a) \otimes \exp(b)) + (\exp(a) \otimes \text{tail}(b) \otimes \exp(b)) \\
&= (\text{tail}(\exp(a)) \otimes \exp(b)) + (\exp(a) \otimes \text{tail}(\exp(b))) \\
&= \text{tail}(\exp(a) \otimes \exp(b)).
\end{aligned}$$

(20)

$$\begin{aligned}
& \text{head}(\sin(a)^2 + \cos(a)^2) = \text{head}(\sin(a)^2) + \text{head}(\cos(a)^2) \\
&= \text{head}(\sin(a) \otimes \sin(a)) + \text{head}(\cos(a) \otimes \cos(a)) \\
&= \text{head}(\sin(a))^2 + \text{head}(\cos(a))^2 = \sin(\text{head}(a))^2 + \cos(\text{head}(a))^2 = 1 = \text{head}(\underline{1}),
\end{aligned}$$

$$\begin{aligned}
tail(\sin(a)^2 + \cos(a)^2) &= tail(\sin(a)^2) + tail(\cos(a)^2) \\
&= tail(\sin(a) \otimes \sin(a)) + tail(\cos(a) \otimes \cos(a)) \\
&= (tail(\sin(a)) \otimes \sin(a)) + (\sin(a) \otimes tail(\sin(a))) \\
&\quad + (tail(\cos(a)) \otimes \cos(a)) + (\cos(a) \otimes tail(\cos(a))) \\
&= (tail(a) \otimes \cos(a) \otimes \sin(a)) + (\sin(a) \otimes tail(a) \otimes \cos(a)) \\
&\quad + (tail(a) \otimes -\sin(a) \otimes \cos(a)) + (\cos(a) \otimes tail(a) \otimes -\sin(a)) \\
&= (tail(a) \otimes \cos(a) \otimes \sin(a)) - (tail(a) \otimes \sin(a) \otimes \cos(a)) \\
&\quad + (\sin(a) \otimes tail(a) \otimes \cos(a)) - (\cos(a) \otimes tail(a) \otimes \sin(a)) \\
&= (\sin(a) \otimes (tail(\sin(a)) + tail(\sin(a)))) + (\cos(a) \otimes (tail(\cos(a)) + tail(\cos(a)))) \\
&= (\sin(a) \otimes ((tail(a) \otimes \cos(a)) + (tail(a) \otimes \cos(a)))) \\
&\quad + (\cos(a) \otimes ((tail(a) \otimes -\sin(a)) + (tail(a) \otimes -\sin(a)))) \\
&= \underline{0} + \underline{0} = \underline{0} = tail(\underline{1})
\end{aligned}$$

(see [48], p. 32).

□

## Fundamental Theorem of Calculus

For all  $f \in \mathbb{A}$ ,

$$f = \lambda x.(f(0) + \int_0^x Df). \quad (21)$$

*Proof.*

$$\begin{aligned} T(\lambda x.(f(0) + \int_0^x Df)) &= T(\lambda x.f(0) + \lambda x. \int_0^x Df) \stackrel{(9)}{=} T(\lambda x.f(0)) + T(\lambda x. \int_0^x Df) \\ &\stackrel{(11)}{=} \underline{f(0)} + (\mathcal{X} \times T(Df)) = \underline{f(0)} + (0 \prec T(Df)) = f(0) \prec T(Df) \\ &= \text{head}(T(f)) \prec \text{tail}(T(f)) = T(f). \end{aligned}$$

Since  $T$  is mono, (21) holds true. □

## Example Fibonacci sequence

$$fib(0) = 0 \wedge fib(1) = 1 \wedge fib = tail(tail(fib)) - tail(fib)$$

$$\iff fib = 0 \prec fib + (1 \prec fib)$$

$$\iff fib = 0 \prec fib' \wedge fib' = 1 \prec fib + fib' \quad ([70], \text{ p. 9})$$

$$\iff fib = \mathcal{X} \times (\underline{1} - \mathcal{X} - \mathcal{X}^2)^{-1} \quad ([148], \text{ p. 111})$$

$$\iff fib = 0 \prec ((\underline{1} + \mathcal{X}) \times fib) + \underline{1} \quad ([48], \text{ p. 33})$$

## Conservative extensions

Let  $\Sigma = (S, F, P)$  be a signature,  $\Sigma' = (S', F', P')$  be a subsignature of  $\Sigma$ ,  $AX$  be a set of  $\Sigma$ -formulas,  $AX' \subseteq AX$  be a set  $\Sigma'$ -formulas,  $A$  be a  $\Sigma$ -algebra and  $B = A|_{\Sigma'}$ .

## Constructor extensions

Let  $\Sigma$  be **constructor** and  $\mu\Sigma$  and  $\mu\Sigma'$  be initial in  $Alg_{\Sigma, AX}$  and  $Alg_{\Sigma', AX'}$ , respectively.

$A$  is  **$F'$ -reachable** (or  **$F'$ -generated**) if  $fold^B : \mu\Sigma' \rightarrow B$  is surjective.

$A$  is **equationally  $F'$ -consistent** if  $fold^B$  is injective.

$(\Sigma, AX)$  is a **conservative extension of**  $(\Sigma', AX')$  if  $\mu\Sigma$  is  $F'$ -reachable and equationally  $F'$ -consistent, i.e., if  $\mu\Sigma|_{\Sigma'}$  and  $\mu\Sigma'$  are isomorphic.

Intuitively,

$A$  is  $F'$ -reachable if each element of  $A$  is obtained by folding an element of  $\mu\Sigma'$ ;

$A$  is equationally  $F'$ -consistent if for each element  $a$  of  $A$  there is only one element of  $\mu\Sigma'$  that folds into  $a$ .

$A$  is  $F'$ -reachable iff  $img(fold^B) = B$ . (1)

$A$  is equationally  $F'$ -consistent iff  $\ker(fold^B) = \Delta_{\mu\Sigma'}$ .

Given a category  $\mathcal{K}$  of  $\Sigma$ -algebras, the full subcategory of  $F$ -reachable objects of  $\mathcal{K}$  is denoted by  $\text{gen}(\mathcal{K})$ .

## Lemma REACH

Let  $A$  be initial in  $\text{Alg}_{\Sigma, AX}$ .

$A$  is  $F'$ -reachable iff  $\text{img}(fold^B)$  is a  $\Sigma$ -invariant.

*Proof.* “ $\Rightarrow$ ”: Let  $A$  be  $F'$ -reachable. Then  $\text{img}(fold^B) = B = A$  and thus  $\text{img}(fold^B)$  is a  $\Sigma$ -invariant.

“ $\Leftarrow$ ”: Let  $\text{img}(fold^B)$  be a  $\Sigma$ -invariant. By Lemma INI (1),  $A$  is the least  $\Sigma$ -invariant of  $A$ . Hence  $B = A \subseteq \text{img}(fold^B) \subseteq B$  and thus by (1),  $A$  is  $F'$ -reachable.  $\square$

## Lemma CONEXT

Let  $\mu\Sigma'$  be extendable to a  $(\Sigma, AX)$ -algebra  $C$ .

Then  $(\Sigma, AX)$  is a conservative extension of  $(\Sigma', AX')$ .

*Proof.* Let  $\text{fold}^C$  be the unique  $\Sigma$ -homomorphism from  $\mu\Sigma$  to  $C$ ,  $A = \mu\Sigma/\ker(\text{fold}^C)$  and  $B = A|_{\Sigma'}$ . By Lemma KER (2), there is a unique  $\Sigma$ -monomorphism  $h : A \rightarrow C$  such that  $(\epsilon)$  commutes:

$$\begin{array}{ccc}
 \mu\Sigma & \xrightarrow{\text{fold}^C} & C \\
 & \searrow \text{nat} & \nearrow h \\
 & (\epsilon) &
 \end{array}$$

By Lemma NAT (4),  $A$  satisfies  $AX$ . Hence  $A \in \text{Alg}_{\Sigma, AX}$  and thus  $B \in \text{Alg}_{\Sigma', AX'}$ . Let  $\text{fold}^B$  be the unique  $\Sigma'$ -homomorphism from  $\mu\Sigma'$  to  $B$ .

$$\mu\Sigma' \xrightarrow{\text{fold}^B} B \xrightarrow{h|_{\Sigma'}} C|_{\Sigma'} = \mu\Sigma'$$

agrees with the identity on  $\mu\Sigma'$  because  $\mu\Sigma'$  is initial. Since  $\text{id}_{\mu\Sigma'}$  is epi, Lemma EPIMON (1) implies that  $h|_{\Sigma'}$  is also epi. We conclude that  $\mu\Sigma'$  and  $B$  are  $\Sigma'$ -isomorphic and thus  $(\Sigma, AX)$  is a conservative extension of  $(\Sigma', AX')$ .  $\square$

## Destructor extensions

Let  $\Sigma$  be **destructor** and  $\nu\Sigma$  and  $\nu\Sigma'$  be final in  $Alg_{\Sigma, AX}$  and  $Alg_{\Sigma', AX'}$ , respectively.

$A$  is  **$F'$ -observable** (or  **$F'$ -cogenerated**) if  $unfold^B : B \rightarrow \nu\Sigma'$  is injective.

$A$  is **behaviorally  $F'$ -complete** if  $unfold^B$  is surjective.

$(\Sigma, AX)$  is a **conservative extension** of  $(\Sigma', AX')$  and  $F \setminus F'$  is **derived from  $F$**  if  $\nu\Sigma$  is  $F'$ -observable and  $F'$ -complete, i.e.,  $\nu\Sigma|_{\Sigma'}$  and  $\nu\Sigma'$  are isomorphic.

Intuitively,

$A$  is  $F'$ -observable if for each element  $a$  of  $A$ , all unfoldings of  $a$  in  $\nu\Sigma'$  are the same;

$A$  is behaviorally  $F'$ -complete if each element of  $\nu\Sigma'$  is the unfolding of an element of  $A$ .

$A$  is  $F'$ -observable iff  $ker(unfold^B) = \Delta_B$ . (3)

$A$  is behaviorally  $F'$ -complete iff  $img(unfold^B) = \nu\Sigma'$ .

Given a category  $\mathcal{K}$  of  $\Sigma$ -algebras, the full subcategory of  $F$ -observable objects of  $\mathcal{K}$  is denoted by  $\text{obs}(\mathcal{K})$ .

## Lemma OBS

Let  $A$  be final in  $\text{Alg}_{\Sigma, AX}$ .

$A$  is  $F'$ -observable iff  $\ker(\text{unfold}^B)$  is a  $\Sigma$ -congruence.

*Proof.* “ $\Rightarrow$ ”: Let  $A$  be  $F'$ -observable. Then  $\ker(\text{unfold}^B) = \Delta_B = \Delta_A$  and thus  $\ker(\text{unfold}^B)$  is a  $\Sigma$ -congruence.

“ $\Leftarrow$ ”: Let  $\ker(\text{unfold}^B)$  be a  $\Sigma$ -congruence. By Lemma FIN (1),  $\Delta_A$  is the greatest  $\Sigma$ -congruence on  $A$ . Hence  $\Delta_B \subseteq \ker(\text{unfold}^B) \subseteq \Delta_A = \Delta_B$  and thus by (3),  $A$  is  $F'$ -observable.  $\square$

## Lemma DESEXT

Let  $\nu\Sigma'$  be extendable to a  $(\Sigma, AX)$ -algebra  $C$ . Then  $(\Sigma, AX)$  is a conservative extension of  $(\Sigma', AX')$ .

*Proof.* Let  $\text{unfold}^C$  be the unique  $\Sigma$ -homomorphism from  $C$  to  $\nu\Sigma$ ,  $A = \text{img}(\text{unfold}^C)$  and  $B = A|_{\Sigma'}$ . By Lemma **IMG** (2), there is a unique  $\Sigma$ -epimorphism  $h : C \rightarrow A$  such that  $(*)$  commutes:

$$\begin{array}{ccc}
 C & \xrightarrow{\text{unfold}^C} & \nu\Sigma \\
 h \searrow & (*) & \swarrow \text{inc} \\
 & A &
 \end{array}$$

By Lemma **INC** (2),  $A$  satisfies  $AX$ . Hence  $A \in \text{Alg}_{\Sigma, AX}$  and thus  $B \in \text{Alg}_{\Sigma', AX'}$ . Let  $\text{unfold}^B$  be the unique  $\Sigma'$ -homomorphism from  $B$  to  $\mu\Sigma'$ .

$$\nu\Sigma' = C|_{\Sigma'} \xrightarrow{h|_{\Sigma'}} B \xrightarrow{\text{unfold}^B} \nu\Sigma'$$

agrees with the identity on  $\nu\Sigma'$  because  $\nu\Sigma'$  is final. Since  $\text{id}_{\nu\Sigma'}$  is mono, Lemma **EPIMON** (2) implies that  $h|_{\Sigma'}$  is also mono. We conclude that  $\nu\Sigma'$  and  $B$  are  $\Sigma'$ -isomorphic and thus  $(\Sigma, AX)$  is a conservative extension of  $(\Sigma', AX')$ . □

## Abstraction

Let  $\Sigma = (S, F, P)$  be a constructor signature,  $\Sigma' = (S, \mathcal{I}, F)$  and  $\mu\Sigma'$  be initial in  $Alg_{\Sigma'}$ .

### Lemma REFL

Let  $h : A \rightarrow B$  be a  $\Sigma$ -homomorphism that preserves all  $p : e \in P$ , i.e.,

$$p^A = \{a \in A_e \mid h(a) \in p^B\},$$

$e = \prod_{x \in V} e_x \in \mathcal{T}_p(S, \mathcal{I})$  and  $\varphi$  be a negation-free  $\Sigma$ -formula over  $V$ .

If  $\varphi$  does not contain universal quantifiers, then

$$h(\varphi^{\mathcal{A}}) \subseteq \varphi^{\mathcal{B}}. \tag{1}$$

If  $h$  is epi, then

$$h^{-1}(\varphi^{\mathcal{B}}) \subseteq \varphi^{\mathcal{A}}. \tag{2}$$

*Proof of (1) by induction on the size of  $\varphi$ .*

Let  $p : e \in P$ ,  $x \in V_s$ . W.l.o.g. we assume that  $r$  is unary.

$$f \in r(t)^A \Leftrightarrow t^A(f) \in r^A \Leftrightarrow t^B(h \circ f) \stackrel{\text{Lemma } \textit{SUBST}}{=} h(t^A(f)) \in r^B \Leftrightarrow h \circ f \in r(t)^B.$$

$$f \in (\varphi \wedge \psi)^A = \varphi^A \cap \psi^A \stackrel{i.h.}{\Rightarrow} h \circ f \in \varphi^B \cap \psi^B = (\varphi \wedge \psi)^B.$$

$$f \in (\varphi \vee \psi)^A = \varphi^A \cup \psi^A \stackrel{i.h.}{\Rightarrow} h \circ f \in \varphi^B \cup \psi^B = (\varphi \vee \psi)^B.$$

$$f \in (\exists x \varphi)^A \Leftrightarrow \exists a \in A_s : upd(f, x, a) \in \varphi^A$$

$$\stackrel{i.h.}{\Rightarrow} \exists a \in A_s : upd(h \circ f, x, h(a)) = h \circ upd(f, x, a) \in \varphi^B$$

$$\Rightarrow \exists b \in B_s : upd(h \circ f, x, b) \in \varphi^B \Leftrightarrow h \circ f \in (\exists x \varphi)^B.$$

*Proof of (2) by induction on the size of  $\varphi$ .*

Let  $r \in R$ ,  $s \in S$  and  $x \in V_s$ . W.l.o.g. we assume that  $r$  is unary.

$$h \circ f \in r(t)^B \Leftrightarrow h(t^A(f)) \stackrel{\text{Lemma } \textit{SUBST}}{=} t^B(h \circ f) \in r^B \Leftrightarrow t^A(f) \in r^A \Leftrightarrow f \in r(t)^A.$$

$$h \circ f \in (\varphi \wedge \psi)^B = \varphi^B \cap \psi^B \stackrel{i.h.}{\Rightarrow} f \in \varphi^A \cap \psi^A = (\varphi \wedge \psi)^A.$$

$$h \circ f \in (\varphi \vee \psi)^B = \varphi^B \cup \psi^B \stackrel{i.h.}{\Rightarrow} f \in \varphi^A \cup \psi^A = (\varphi \vee \psi)^A.$$

$$h \circ f \in (\exists x \varphi)^B \Leftrightarrow \exists b \in B_s : upd(h \circ f, x, b) \in \varphi^B$$

$$\stackrel{h \text{ epi}}{\Rightarrow} \exists a \in A_s : h \circ upd(f, x, a) = upd(h \circ f, x, h(a)) \in \varphi^B$$

$$\stackrel{i.h.}{\Rightarrow} \exists a \in A_s : upd(f, x, a) \in \varphi^A \Leftrightarrow f \in (\exists x \varphi)^A.$$

$$h \circ f \in (\forall x \varphi)^B \Leftrightarrow \forall b \in B_s : upd(h \circ f, x, b) \in \varphi^B$$

$$\Rightarrow \forall a \in A_s : h \circ upd(f, x, a) = upd(h \circ f, x, h(a)) \in \varphi^B$$

$$\stackrel{i.h.}{\Rightarrow} \forall a \in A_s : upd(f, x, a) \in \varphi^A \Leftrightarrow f \in (\forall x \varphi)^A. \quad \square$$

## Abstraction with a least congruence

Let  $AX$  consist of  $\forall$ -free Horn clauses and  $\mathcal{K} = \text{Alg}_{\Sigma, AX}$  such that for all  $A \in \mathcal{K}$ ,  $=^A$  is a  $\Sigma$ -congruence, and  $C = \text{lfp}(\mu\Sigma', \Sigma, AX)$ .

Then  $\sim =_{def} =^C$  is the least  $\Sigma$ -congruence on  $\mu\Sigma'$ .

By Lemma NAT,  $C/\sim \in \mathcal{K}$ .

Let  $A \in \mathcal{K}$ . We define  $B \in \text{Alg}_{\Sigma}$  as the  $\text{fold}^A$ -pre-image of the interpretation of  $R$  in  $A$ , i.e., for all  $r : w \in R$ ,

$$r^B =_{def} \{b \in \mu\Sigma'_w \mid \text{fold}^A(b) \in r^A\}.$$

Use induction on  $\mathbb{N}$  and Kleene's Fixpoint Theorem (or transfinite induction and Zermelo's Fixpoint Theorem) to show that  $\text{fold}^A$  extends to a  $\Sigma$ -homomorphism!

$B$  satisfies  $AX$  and thus  $B \in \text{Alg}_{\Sigma, AX}$ .

*Proof.* Let  $\varphi = (r(t_1, \dots, t_n) \Leftarrow \psi) \in AX$  and  $g \in \psi^B$ . By Lemma **REFL** (1),  $\text{fold}^A \circ g \in \psi^A$ . Since  $A$  satisfies  $\varphi$ ,  $\text{fold}^A \circ g \in r(t_1, \dots, t_n)^A$ , i.e.,

$$(\text{fold}^A(t_1^B(g)), \dots, \text{fold}^A(t_n^B(g))) \stackrel{\text{Lemma } \text{SUBST}}{=} (t_1^A(\text{fold}^A \circ g), \dots, t_n^A(\text{fold}^A \circ g)) \in r^A.$$

Hence  $(t_1^B(g), \dots, t_n^B(g)) \in r^B$  and thus  $g \in r(t_1, \dots, t_n)^B$ . □

**Theorem ABSINI**  $C/\sim$  is initial in  $\mathcal{K}$ .

*Proof.* Since  $C$  is the least  $D \in \text{Alg}_{\Sigma, AX}$  with  $D|_{\Sigma'} = \mu\Sigma'$ , we obtain  $C \leq B$ . In particular,

$$\begin{aligned} \sim &= =^C \subseteq =^B = \{(t, u) \in (\mu\Sigma')^2 \mid \text{fold}^A(t) =^A \text{fold}^A(u)\} \\ &= \text{ker}(\text{fold}^A) \end{aligned}$$

because  $=^A = \Delta_A$ . Hence  $h : C/\sim \rightarrow A$  is well-defined by  $h \circ \text{nat}_\sim = \text{fold}^A \circ \text{id}_{\mu\Sigma'}$ .

$$\begin{array}{ccc}
 C & \xrightarrow{\textcolor{blue}{nat}_{\sim}} & C/\sim \\
 id_{\mu\Sigma'} \downarrow & & \downarrow h \\
 B & \xrightarrow{\textcolor{blue}{fold}^A} & A
 \end{array}$$

Since  $\text{nat}_{\sim}$  is epi and reflects predicates and  $\text{fold}^A \circ id_{\mu\Sigma'}$  is  $\Sigma$ -homomorphic, Lemma **EMH** (1) implies that  $h$  is also  $\Sigma$ -homomorphic.

Let  $h'$  be any  $\Sigma$ -homomorphism from  $C/\sim$  to  $A$ . Since  $B|_{B\Sigma} = BA$  is initial in  $\text{Alg}_{\Sigma}$ ,  $h' \circ \text{nat}_{\sim} = h \circ \text{nat}_{\sim}$  and thus  $h' = h$  because  $\text{nat}_{\sim}$  is epi. □

## Abstraction with a greatest congruence

Let  $AX$  consist of co-Horn clauses and  $\mathcal{K} = \text{Alg}_{\Sigma, AX}$  such that for all  $A \in \mathcal{K}$ ,  $=^A$  is a  $\Sigma$ -congruence,  $C = \text{gfp}(\mu\Sigma', \Sigma, AX)$  and  $\sim =_{\text{def}} =^C$  be a  $\Sigma$ -congruence on  $\mu\Sigma'$ . Hence  $C \in \text{gen}(\mathcal{K})$ .

By Lemma NAT,  $C/\sim \in \text{gen}(\mathcal{K})$ .

Let  $A \in \text{gen}(\mathcal{K})$ . We define  $B \in \text{Alg}_{\Sigma}$  as the  $\text{fold}^A$ -pre-image of the interpretation of  $R$  in  $A$ , i.e., for all  $r : w \in R$ ,

$$r^B =_{\text{def}} \{b \in \mu\Sigma'_w \mid \text{fold}^A(b) \in r^A\}.$$

Use induction on  $\mathbb{N}$  and Kleene's Fixpoint Theorem (or transfinite induction and Zermelo's Fixpoint Theorem) to show that  $\text{fold}^A$  extends to a  $\Sigma$ -homomorphism!

$B$  satisfies  $AX$  and thus  $B \in \text{gen}(\mathcal{K})$ .

*Proof.* Let  $r \in R$ ,  $\varphi = (r(t_1, \dots, t_n) \Rightarrow \psi) \in AX$  and  $g \in r(t_1, \dots, t_n)^B$ . Hence  $(t_1^B(g), \dots, t_n^B(g)) \in r^B$  and thus

$$(t_1^A(\text{fold}^A \circ g), \dots, t_n^A(\text{fold}^A \circ g)) \stackrel{\text{Lemma SUBST}}{=} (\text{fold}^A(t_1^B(g)), \dots, \text{fold}^A(t_n^B(g))) \in r^A.$$

Hence  $\text{fold}^A \circ g \in r(t_1, \dots, t_n)^A$ . Since  $A$  satisfies  $\varphi$ ,  $\text{fold}^A \circ g \in \psi^A$ . Since  $A$  is  $\Sigma$ -reachable,  $\text{fold}^A$  is epi and thus Lemma REFL (2) implies  $g \in \psi^B$ .  $\square$

**Theorem ABSFIN**  $C/\sim$  is final in  $\text{gen}(\mathcal{K})$ .

*Proof.* Since  $C$  is the greatest  $D \in \text{Alg}_{\Sigma, AX}$  with  $D|_{B\Sigma} = \mu\Sigma'$ , we obtain  $B \leq C$ . In particular,

$$\ker(\text{fold}^A) = \{(t, u) \in (\mu\Sigma')^2 \mid \text{fold}^A(t) =^A \text{fold}^A(u)\} = =^B \subseteq =^C = \sim$$

because  $=^A = \Delta_A$ .

Hence for all  $t, u \in \mu\Sigma'$ ,  $\text{fold}^A(t) = \text{fold}^A(u)$  implies  $t \sim u$ . Since  $A$  is  $\Sigma$ -reachable,  $\text{fold}^A$  is epi and thus for all  $a \in A$  there is  $t \in \mu\Sigma'$  with  $\text{fold}^A(t) = a$ .

Hence  $h : A \rightarrow C/\sim$  is well-defined by  $h \circ fold^A = nat_\sim \circ id_{\mu\Sigma'}$ .

$$\begin{array}{ccc}
 B & \xrightarrow{\textcolor{blue}{fold}^A} & A \\
 \downarrow id_{\mu\Sigma'} & & \downarrow h \\
 C & \xrightarrow{\textcolor{blue}{nat}_\sim} & C/\sim
 \end{array}$$

Since  $fold^A$  is epi and reflects predicates and  $nat_\sim \circ id_{\mu\Sigma'}$  is  $\Sigma$ -homomorphic, Lemma EMH (1) implies that  $h$  is also  $\Sigma_{BA'}$ -homomorphic.

Let  $h'$  be any  $\Sigma$ -homomorphism from  $A$  to  $C/\sim$ . Since  $B|_{B\Sigma} = \nu\Sigma'$  is initial in  $Alg_\Sigma$ ,  $h' \circ fold^A = h \circ fold^A$  and thus  $h' = h$  because  $fold^A$  is epi.  $\square$

## Restriction

Let  $\Sigma = (S, F, P)$  be a destructor signature,  $\Sigma' = (S, \mathcal{I}, F)$  and  $\nu\Sigma'$  be final in  $Alg_{\Sigma'}$ .

### Lemma PRES

Let  $h : A \rightarrow B$  be a  $\Sigma$ -homomorphism that preserves all  $p \in P$ , i.e.,

$$p^B = h(p^A)$$

and  $\varphi$  be a negation-free  $\Sigma$ -formula.

If  $\varphi$  does not contain universal quantifiers, then

$$f \in \varphi^A \text{ implies } h \circ f \in \varphi^B. \quad (3)$$

If  $h$  is mono and for all atomic subformulas  $r(t_1, \dots, t_n)$  of  $\varphi$ ,  $t_1, \dots, t_n$  are variables, then

$$g \in \varphi^B \text{ implies } \exists f \in \varphi^A : h \circ f =_{free(\varphi)} g. \quad (4)$$

Let  $r \in R$ ,  $s \in S$  and  $x \in V_s$ . W.l.o.g. we assume that  $r$  is unary.

*Proof of (3) by induction on the size of  $\varphi$ .*

$$f \in r(t)^A \Leftrightarrow t^A(f) \in r^A \Leftrightarrow t^B(h \circ f) \stackrel{\text{Lemma SUBST}}{=} h(t^A(f)) \in r^B \Leftrightarrow h \circ f \in r(t)^B.$$

$$f \in (\varphi \wedge \psi)^A = \varphi^A \cap \psi^A \stackrel{i.h.}{\Rightarrow} h \circ f \in \varphi^B \cap \psi^B = (\varphi \wedge \psi)^B.$$

$$f \in (\varphi \vee \psi)^A = \varphi^A \cup \psi^A \stackrel{i.h.}{\Rightarrow} h \circ f \in \varphi^B \cup \psi^B = (\varphi \vee \psi)^B.$$

$$\begin{aligned} f \in (\exists x \varphi)^A &\Leftrightarrow \exists a \in A_s : \text{upd}(f, x, a) \in \varphi^A \\ &\stackrel{i.h.}{\Rightarrow} \exists a \in A_s : \text{upd}(h \circ f, x, h(a)) = h \circ \text{upd}(f, x, a) \in \varphi^B \\ &\Rightarrow \exists b \in B_s : \text{upd}(h \circ f, x, b) \in \varphi^B \Leftrightarrow h \circ f \in (\exists x \varphi)^B. \end{aligned}$$

*Proof of (4) by induction on the size of  $\varphi$ .*

Let  $r \in R$ ,  $s \in S$  and  $x \in V_s$ . W.l.o.g. we assume that  $r$  is unary.

$$\begin{aligned} g \in r(z)^B &\Leftrightarrow g(z) \in r^B \Leftrightarrow \exists a \in r^A : h(a) = g(z) \\ &\Leftrightarrow \exists f \in A^X : f(z) \in r^A \wedge h \circ f =_{\{z\}} g \Leftrightarrow \exists f \in r(z)^A : h \circ f =_{\text{free}(r(z))} g. \\ g \in (\varphi \wedge \psi)^B &= \varphi^B \cap \psi^B \stackrel{i.h.}{\Rightarrow} \exists f \in \varphi^A : h \circ f =_{\text{free}(\varphi)} g \wedge \exists f' \in \psi^A : h \circ f' =_{\text{free}(\psi)} g \\ &\stackrel{h \text{ mono}}{\Rightarrow} \exists f \in \varphi^A \cap \psi^A : h \circ f =_{\text{free}(\varphi) \cup \text{free}(\psi)} g \\ &\Leftrightarrow \exists f \in (\varphi \wedge \psi)^A : h \circ f =_{\text{free}(\varphi \wedge \psi)} g. \end{aligned}$$

$$g \in (\varphi \vee \psi)^B \stackrel{\text{analogously}}{\Rightarrow} \exists f \in (\varphi \vee \psi)^A : h \circ f = g.$$

$$g \in (\exists x \varphi)^B \Leftrightarrow \exists b \in B_s : upd(g, x, b) \in \varphi^B$$

$$\stackrel{i.h.}{\Rightarrow} \exists b \in B_s : \exists f \in \varphi^A : h \circ f =_{free(\varphi)} upd(g, x, b)$$

$$\Rightarrow \exists f \in A^X : \exists a \in A_s : upd(f, x, a) \in \varphi^A \wedge h \circ f =_{free(\varphi) \setminus \{x\}} g$$

$$\Rightarrow \exists f \in (\exists x \varphi)^A : h \circ f =_{free(\exists x \varphi)} g.$$

$$g \in (\forall x \varphi)^B \Leftrightarrow \forall b \in B_s : upd(g, x, b) \in \varphi^B$$

$$\stackrel{i.h.}{\Rightarrow} \forall b \in B_s : \exists f \in \varphi^A : h \circ f =_{free(\varphi)} upd(g, x, b)$$

$$\stackrel{h \text{ mono}}{\Rightarrow} \exists f \in A^X : \forall a \in A_s : upd(f, x, a) \in \varphi^A \wedge h \circ f =_{free(\varphi) \setminus \{x\}} g$$

$$\Rightarrow \exists f \in (\forall x \varphi)^A : h \circ f =_{free(\forall x \varphi)} g. \quad \square$$

## Restriction with a greatest invariant

Let  $AX$  consist of co-Horn clauses  $r(t_1, \dots, t_n) \Rightarrow \psi$  and  $\mathcal{K} = \text{Alg}_{\Sigma, AX}$  such that for all  $A \in \mathcal{K}$ ,  $\in^A$  is a  $\Sigma$ -invariant,  $t_1, \dots, t_n$  are variables,  $\text{free}(\psi) \subseteq \{t_1, \dots, t_n\}$  and  $\psi$  is  $\forall$ -free and constraint compatible. Let  $C = \text{gfp}(\nu\Sigma', \Sigma, AX)$ . Then  $\text{inv} = \in^C$  is the greatest  $\Sigma$ -invariant of  $\nu\Sigma'$ .

By Lemma INC,  $\text{inv} \in \mathcal{K}$ .

Let  $A \in \mathcal{K}$ . We define  $B \in \text{Alg}_\Sigma$  as the  $\text{unfold}^A$ -image of the interpretation of  $R$  in  $A$ , i.e., for all  $r \in R$ ,

$$r^B =_{\text{def}} \text{unfold}^A(r^A).$$

Use induction on  $\mathbb{N}$  and Kleene's Fixpoint Theorem (or transfinite induction and Zermelo's Fixpoint Theorem) to show that  $\text{unfold}^A$  extends to a  $\Sigma$ -homomorphism!

$B$  satisfies  $AX$  and thus  $B \in \mathcal{K}$ .

*Proof.* W.l.o.g. let  $\varphi = (r(x_1, \dots, x_n) \Rightarrow \psi) \in AX$  and  $g \in r(x_1, \dots, x_n)^B$ . Hence  $(g(x_1), \dots, g(x_n)) \in r^B$  and thus  $(f(x_1), \dots, f(x_n)) \in r^A$  and  $\text{unfold}^A \circ f =_{\{x_1, \dots, x_n\}} g$  for some  $f \in A^X$ . Hence  $f \in \psi^A$  because  $A$  satisfies  $\varphi$ , and thus by Lemma PRES (1),  $\text{unfold}^A \circ f \in \psi^B$ . Therefore,  $\text{free}(\psi) \subseteq \{x_1, \dots, x_n\}$  implies  $g \in \psi^B$ .  $\square$

**Theorem RESFIN**  $inv$  is final in  $\mathcal{K}$ .

*Proof.* Since  $C$  is the greatest  $D \in Alg_{\Sigma, AX}$  with  $D|_{\Sigma'} = \nu\Sigma'$ , we obtain  $B \leq C$ . In particular,

$$\begin{aligned} img(unfold^A) &= \{unfold^A(a) \mid a \in A\} = \{unfold^A(a) \mid a \in mem^A\} \\ &= \underline{\in^B} \subseteq \underline{\in^C} = inv \end{aligned}$$

because  $\underline{\in^A} = A$ . Hence  $h : A \rightarrow inv$  is well-defined by  $inc \circ h = id_{\nu\Sigma'} \circ unfold^A$ .

$$\begin{array}{ccc} inv & \xrightarrow{inc} & C \\ h \uparrow & & \downarrow id_{\nu\Sigma'} \\ A & \xrightarrow{unfold^A} & B \end{array}$$

Since  $inc$  is mono and reflects predicates and  $id_{\nu\Sigma'} \circ unfold^A$  is  $\Sigma$ -homomorphic, Lemma EMH (2) implies that  $h$  is also  $\Sigma$ -homomorphic.

Let  $h'$  be any  $\Sigma$ -homomorphism from  $A$  to  $inv$ . Since  $B|_{\Sigma'} = \nu\Sigma'$  is final in  $Alg_{\Sigma}$ ,  $inc \circ h' = inc \circ h$  and thus  $h' = h$  because  $inc$  is mono.  $\square$

## Example Length of a colist

Let  $\Sigma = (S, F' \cup \{length : list \rightarrow nat\}, \{\in : list\})$ ,  $\Sigma' = (S, F' \cup \{length\}, \emptyset)$  and  $AX$  be a set of co-Horn clauses such that for all  $A \in Alg_{\Sigma, AX}$ ,  $p_{list}^A$  is a  $\Sigma$ -invariant, and  $AX$  includes the following co-Horn clauses:

$$p_{list}(s) \Rightarrow length(s) = \lambda\{\iota_1.zero, \iota_2(x, s).succ(length(s))\}(split(s)).$$

Let  $A = gfp(\Sigma, \nu\Sigma', AX)$ . By Theorem RESFIN,  $\in^A$  is final in  $Alg_{\Sigma, AX}$ . Since the final  $coList(X)$ -algebra is a  $(\Sigma, AX)$ -algebra, we conclude from Lemma DESEXT that  $(\Sigma, AX)$  is a conservative extension of  $(coList(X), \emptyset)$ .  $\square$

## Example Subtree of a cobintree

Let  $\Sigma = (S, F' \cup \{subtree' : btree \rightarrow (2^* \rightarrow btree)\}, \{\in : btree\})$ ,  $\Sigma' = (S, F' \cup \{subtree'\}, \emptyset)$  and  $AX$  be a set of co-Horn clauses such that for all  $A \in Alg_{\Sigma, AX}$ ,  $p_{btree}^A$  is a  $\Sigma$ -invariant, and  $AX$  includes the following co-Horn clauses:

$$p_{btree}(t) \Rightarrow subtree'(t)(\epsilon) = t,$$

$$p_{btree}(t) \Rightarrow (split(t) = (x, u, u') \Rightarrow subtree'(t)(0:w) = subtree'(u)(w)),$$

$$p_{btree}(t) \Rightarrow (split(t) = (x, u, u') \Rightarrow subtree'(t)(1:w) = subtree'(u')(w)).$$

Let  $A = gfp(\Sigma, \nu\Sigma', AX)$ . By Theorem **RESFIN**,  $\in^A$  is final in  $Alg_{\Sigma, AX}$ . Since the final  $coBintree(X)$ -algebra is a  $(\Sigma, AX)$ -algebra, we conclude from Lemma **DESEXT** that  $(\Sigma, AX)$  is a conservative extension of  $(coBintree(X), \emptyset)$ .  $\square$

### Example Infinite trees, AG and EG (see chapter 9)

Let  $\Sigma = (S, F', \{infinite, AG, EG\})$  and  $AX$  be set of co-Horn clauses such that for all  $A \in Alg_{\Sigma, AX}$ ,  $\in^A$  is a  $\Sigma$ -invariant. Moreover, let  $AX$  include the following axioms:

$$infinite(t) \Rightarrow \exists x, u, u' : split(t) = (x, u, u') \wedge (infinite(u) \vee infinite(u'))$$

$$AG(P)(t) \Rightarrow \exists x, u, u' : (split(t) = (x, u, u') \Rightarrow (P(x) \wedge AG(P)(u) \wedge AG(P)(u'))))$$

$$EG(P)(t) \Rightarrow \exists x, u, u' : (split(t) = (x, u, u') \Rightarrow (P(x) \wedge (EG(P)(u) \vee EG(P)(u')))))$$

where  $P$  is a predicate variable.

Let  $A = \text{lfp}(\nu \text{coBintree}, \Sigma, AX)$ . By Theorem RESFIN,  $\in^A$  is final in  $\text{Alg}_{\Sigma, AX}$ , the category of  $\Sigma$ -algebras  $B$  such that  $B$  satisfies  $AX$  and  $\in^B = B$ .  $\square$

## Restriction with a least invariant

Let  $AX$  consist of Horn clauses  $r(t_1, \dots, t_n) \Leftarrow \psi$  and  $\mathcal{K} = \text{Alg}_{\Sigma, AX}$  such that for all  $A \in \mathcal{K}$ ,  $\in^A$  is a  $\Sigma$ -invariant,  $\text{free}(r(t_1, \dots, t_n)) \subseteq \text{free}(\psi)$ ,  $\psi$  is constraint compatible and for all atomic subformulas  $p(u_1, \dots, u_m)$  of  $\psi$ ,  $u_1, \dots, u_m$  are variables. Let  $C = \text{lfp}(\nu \Sigma', \Sigma, AX)$  and  $\text{inv} = \in^C$  be a  $\Sigma$ -invariant of  $\nu \Sigma'$ . Hence  $C \in \text{obs}(\mathcal{K})$ .

By Lemma INC,  $\text{inv} \in \text{obs}(\mathcal{K})$ .

Let  $A \in \mathcal{K}$ . We define  $B \in \text{Alg}_{\Sigma}$  as the  $\text{unfold}^A$ -image of the interpretation of  $R$  in  $A$ , i.e., for all  $r \in R$ ,

$$r^B =_{\text{def}} \text{unfold}^A(r^A).$$

Use induction on  $\mathbb{N}$  and Kleene's Fixpoint Theorem (or transfinite induction and Zermelo's Fixpoint Theorem) to show that  $\text{unfold}^A$  extends to a  $\Sigma$ -homomorphism!

$B$  satisfies  $AX$  and thus  $B \in obs(\mathcal{K})$ .

*Proof.* Let  $\varphi = (r(t_1, \dots, t_n) \Leftarrow \psi) \in AX$  and  $g \in \psi^B$ . Since  $A$  is  $\Sigma$ -observable,  $unfold^A$  is mono and thus Lemma PRES (2) implies  $g =_{free(\psi)} unfold^A \circ f$  for some  $f \in \psi^A$ . Since  $A$  satisfies  $\varphi$ ,  $f \in r(t_1, \dots, t_n)^A$  and thus  $(t_1^A(f), \dots, t_n^A(f)) \in r^A$ . Hence

$$\begin{aligned} & (t_1^B(unfold^A \circ f), \dots, t_n^B(unfold^A \circ f)) \\ & \stackrel{\text{Lemma SUBST}}{=} (unfold^A(t_1^A(f)), \dots, unfold^A(t_n^A(f))) \in r^B \end{aligned}$$

and thus  $unfold^A \circ f \in r(t_1, \dots, t_n)^B$ . Therefore,  $free(r(t_1, \dots, t_n)) \subseteq free(\psi)$  implies  $g \in r(t_1, \dots, t_n)^B$ .  $\square$

**Theorem RESINI**  $inv$  is initial in  $obs(\mathcal{K})$ .

*Proof.* Since  $C$  is the least  $D \in \mathcal{K}$  with  $D|_{\Sigma'} = \nu\Sigma'$ , we obtain  $C \leq B$ . In particular,

$$\begin{aligned} inv &= \in^C \subseteq \in^B = \{unfold^A(a) \mid a \in mem^A\} = \{unfold^A(a) \mid a \in A\} \\ &= img(unfold^A) \end{aligned} \tag{*}$$

because  $\in^A = A$ . Since  $A$  is  $\Sigma$ -observable,  $unfold^A$  is mono and thus for all  $a, b \in A$ ,  $unfold^A(a) = unfold^A(b)$  implies  $a = b$ .

Hence by (\*),  $h : \text{inv} \rightarrow A$  with  $h(b) = (\text{unfold}^A)^{-1}(b)$  for all  $b \in \text{inv}$  is well-defined. Therefore,  $\text{unfold}^A \circ h = \text{id}_{\nu\Sigma'} \circ \text{inc}$ .

$$\begin{array}{ccc}
 A & \xrightarrow{\text{unfold}^A} & B \\
 \downarrow h & & \downarrow \text{id}_{\nu\Sigma'} \\
 \text{inv} & \xrightarrow{\text{inc}} & C
 \end{array}$$

Since  $\text{unfold}^A$  is mono and reflects predicates and  $\text{id}_{\nu\Sigma'} \circ \text{inc}$  is  $\Sigma$ -homomorphic, Lemma EMH (2) implies that  $h$  is also  $\Sigma$ -homomorphic.

Let  $h'$  be any  $\Sigma$ -homomorphism from  $\text{inv}$  to  $A$ . Since  $B|_{B\Sigma} = BA$  is final in  $\text{Alg}_\Sigma$ ,  $\text{unfold}^A \circ h' = \text{unfold}^A \circ h$  and thus  $h' = h$  because  $\text{unfold}^A$  is mono.  $\square$

### Example Finite trees, EF and AF (see chapter 9)

Let  $\Sigma = (S, F', \{\text{finite}, \text{EF}, \text{AF}\})$  and  $AX$  be a set of  $\Sigma$ -Horn clauses such that for all  $A \in \text{Alg}_{\Sigma, AX}$ ,  $\in^A$  is a  $\Sigma$ -invariant. Moreover, let  $AX$  include the following axioms:

$$\text{finite}(t) \Leftarrow \text{split}(t) = \epsilon \vee (\text{split}(t) = (x, u, u') \wedge \text{finite}(u) \wedge \text{finite}(u'))$$

$$EF(P)(t) \Leftarrow \text{split}(t) = (x, u, u') \wedge (P(x) \vee EF(P)(u) \vee EF(u'))$$

$$AF(P)(t) \Leftarrow \text{split}(t) = (x, u, u') \wedge (P(x) \vee (AF(P)(u) \wedge AF(u')))$$

where  $P$  is a predicate variable.

Let  $A = \text{lfp}(\Sigma, \nu \text{coBintree}, AX)$ . By Theorem RESINI,  $\in^A$  is initial in  $\text{obs}(\text{Alg}_{\Sigma, AX})$ , the category of  $F'$ -observable  $\Sigma$ -coalgebras  $B$  such that  $B$  satisfies  $AX$  and  $\in^B = B$ .  $\square$

## Example Cotrees with finite outdegree

Let  $AX$  be given by the following Horn clauses over  $\text{coTree}$ :

$$\text{is}_{\text{tree}}(t) \Leftarrow \text{is}_{\text{trees}}(\text{subtrees}\langle t \rangle)$$

$$\text{is}_{\text{trees}}(ts) \Leftarrow [[x, y]\text{split}]ts = [x]p \vee$$

$$([[x, y]\text{split}]ts = [y]p \wedge \text{is}_{\text{tree}}(\pi_1\langle p \rangle) \wedge \text{is}_{\text{trees}}(\pi_2\langle p \rangle))$$

$AX$  satisfies the assumptions of Restriction with a least invariant. Hence  $\text{inv} = \in^{\text{lfp}(\overline{AX})}$  is initial in  $\text{obs}(\text{Alg}_{\text{coTree}, AX})$ , the category of  $\text{coTree}$ -observable  $\text{coTree}$ -coalgebras  $A$  such that  $A$  satisfies  $AX$  and  $\in^A = A$ .  $\square$

## $\lambda$ -bialgebras

Given two endofunctors  $T, D$  on  $\mathcal{K}$ , a **distributive law** of  $T$  over  $D$  is a natural transformation  $\lambda : TD \rightarrow DT$ .

Given a distributive law  $\lambda$  of  $T$  over  $D$ , a pair of  $\mathcal{K}$ -morphisms

$$(\alpha : TA \rightarrow A, \beta : A \rightarrow DA),$$

is a  **$\lambda$ -bialgebra** if the following diagram, called **pentagonal law**, commutes:

$$\begin{array}{ccccc}
 TA & \xrightarrow{\alpha} & A & \xrightarrow{\beta} & DA \\
 \downarrow T\beta & & & & \uparrow D\alpha \\
 TDA & \xrightarrow{\lambda_A} & DTA & &
 \end{array}$$

$\lambda$  lifts  $T$  and  $D$  to endofunctors on  $coAlg_T$  and  $Alg_T$ , respectively (see, e.g., [26, 86, 70]):

$$\begin{aligned}
 T_\lambda : coAlg_D &\rightarrow coAlg_D \\
 A \xrightarrow{\beta} DA &\mapsto TA \xrightarrow{T\beta} TDA \xrightarrow{\lambda_A} DTA \\
 h : A \rightarrow B &\mapsto Th : TA \rightarrow TB
 \end{aligned}$$

$D^\lambda : Alg_T \rightarrow Alg_T$

$TA \xrightarrow{\alpha} A \mapsto TDA \xrightarrow{\lambda_A} DTA \xrightarrow{D\alpha} DA$

$h : A \rightarrow B \mapsto Dh : DA \rightarrow DB$

Hence the pentagonal law can also be written as the following square (1), which shows that  $\alpha$  is a  $coAlg_D$ -morphism from  $T_\lambda\beta$  to  $\beta$ , or as the square (2), which shows that  $\beta$  is an  $Alg_T$ -morphism from  $\alpha$  to  $D^\lambda\alpha$ :

$$\begin{array}{ccc}
 TA & \xrightarrow{\alpha} & A \\
 \downarrow T_\lambda\beta & (1) & \downarrow \beta \\
 DTA & \xrightarrow{D\alpha} & DA
 \end{array}
 \qquad
 \begin{array}{ccc}
 A & \xrightarrow{\beta} & DA \\
 \uparrow \alpha & (2) & \uparrow D^\lambda\alpha \\
 TA & \xrightarrow{T\beta} & TDA
 \end{array}$$

$biAlg_\lambda$  denotes the category of  $\lambda$ -bialgebras and  $biAlg_\lambda$ -morphisms:

A  $biAlg_\lambda$ -morphism  $h$  from a  $\lambda$ -bialgebra  $TA \xrightarrow{\alpha} A \xrightarrow{\beta} DA$  to a  $\lambda$ -bialgebra  $TB \xrightarrow{\gamma} B \xrightarrow{\delta} DB$  is a  $\mathcal{K}$ -morphism  $h : A \rightarrow B$  such that the following diagrams commute:  $h \circ \alpha = \gamma \circ Th$  and  $Dh \circ \beta = \delta \circ h$ .

$$\begin{array}{ccccc}
 TA & \xrightarrow{\alpha} & A & \xrightarrow{\beta} & DA \\
 \downarrow Th & & \downarrow h & & \downarrow Dh \\
 TB & \xrightarrow{\gamma} & B & \xrightarrow{\delta} & DB
 \end{array}$$

Bialgebras generalize both algebras and coalgebras: Let  $id_T, id_D$  be the identity transformations on  $T, D$ , respectively, and  $Id_{\mathcal{K}}$  be the identity functor on  $\mathcal{K}$ .  $id_T$  is a distributive law of  $T$  over  $Id_{\mathcal{K}}$ , while  $id_D$  is a distributive law of  $Id_{\mathcal{K}}$  over  $D$ . Hence every  $T$ -algebra is an  $id_T$ -bialgebra and every  $D$ -coalgebra is an  $id_D$ -bialgebra.

## Lemma BINIFIN

(3) Let  $T(\mu T) \xrightarrow{\alpha} \mu T$  be initial in  $Alg_T$ . Then

$$T(\mu T) \xrightarrow{\alpha} \mu T \xrightarrow{fold^{D\lambda_\alpha}} D(\mu T)$$

is initial in  $biAlg_\lambda$ . Conversely, all solutions  $\beta$  of the pentagonal law with  $A = \mu T$  agree with  $fold^{D\lambda_\alpha}$ .

(4) Let  $\nu D \xrightarrow{\beta} D(\nu D)$  be final in  $coAlg_D$ . Then

$$T(\nu D) \xrightarrow{unfold^{T\lambda\beta}} \nu D \xrightarrow{\beta} D(\nu D)$$

is final in  $biAlg_\lambda$ . Conversely, all solutions  $\alpha$  of the pentagonal law with  $A = \nu D$  agree with  $unfold^{T\lambda\beta}$ .

*Proof.* See [86], section 4; or [70], section 6. The second part of (3) follows from the fact that the pentagonal law commutes iff diagram (2) with  $A = \mu T$  commutes iff  $\beta$  is an  $Alg_T$ -morphism. The second part of (4) follows from the fact that the pentagonal law commutes iff diagram (1) with  $A = \nu D$  commutes iff  $\alpha$  is a  $coAlg_D$ -morphism.  $\square$

In particular, there is a unique  $biAlg_\lambda$ -morphism  $h$  from the initial  $\lambda$ -bialgebra to the final one:

$$\begin{array}{ccccc}
 T(\mu T) & \xrightarrow{\alpha} & \mu T & \xrightarrow{fold^{D^{\lambda\alpha}}} & D(\mu T) \\
 Th \downarrow & (5) & \downarrow h & (6) & \downarrow Dh \\
 T(\nu D) & \xrightarrow{unfold^{T\lambda\beta}} & \nu D & \xrightarrow{\beta} & D(\nu D)
 \end{array}$$

Since  $unfold^{T\lambda\beta}$  equips  $\nu D$  with a  $T$ -algebra structure and  $\mu T$  is the initial  $T$ -algebra,  $h = fold^{\nu D}$ . Since  $fold^{D^{\lambda\alpha}}$  equips  $\mu T$  with a  $D$ -coalgebra structure and  $\nu D$  is the final  $D$ -coalgebra,  $h = unfold^{\mu T}$ . Hence  $fold^{\nu D} = h = unfold^{\mu T}$ .

## Monads and comonads

A **monad** (or **algebraic theory in monoid form**) in  $\mathcal{K}$  is a triple  $M = (T, \eta, \mu)$  consisting of a functor  $T : \mathcal{K} \rightarrow \mathcal{K}$  and natural transformations  $\eta : Id_{\mathcal{K}} \rightarrow T$  (**unit**) and  $\mu : TT \rightarrow T$  (**multiplication**) such that the following diagrams commute:

$$\begin{array}{ccccc}
 & T & \xrightarrow{\eta T} & TT & \xleftarrow{T\eta} T \\
 & id_T \searrow & \downarrow \mu & \nearrow id_T & \\
 & & T & &
 \end{array}
 \qquad
 \begin{array}{ccc}
 TTT & \xrightarrow{\mu T} & TT \\
 T\mu \downarrow & & \downarrow \mu \\
 TT & \xrightarrow{\mu} & T
 \end{array}$$

If  $\eta$  and  $\mu$  are clear from the context,  $M$  is identified with  $T$ .

A monad in  $\mathcal{K}$  is a monoid in the category  $\mathcal{K}^{\mathcal{K}}$  with functors as objects and natural transformations as morphisms.

A **Kleisli triple**  $(T, \eta, \_)^*$  consists of a function  $T : \mathcal{K} \rightarrow \mathcal{K}$ , sets

$\eta = \{\eta_A : A \rightarrow TA \mid A \in \mathcal{K}\}$  and  $\_^* = \{f^* : TA \rightarrow TB \mid A, B \in \mathcal{K}, f : A \rightarrow TB\}$  such that for all  $A \in \mathcal{K}$ ,  $f : A \rightarrow TB$  and  $g : B \rightarrow TC$ ,

$$\eta_A^* = id_{TA}, \quad f^* \circ \eta_A = f, \quad (g^* \circ f)^* = g^* \circ f^*.$$

A Kleisli triple  $(T, \eta, \_)^*$  defines the monad  $(T, \eta, \mu)$  with  $Tf = (\eta_B \circ f)^*$  and  $\mu_A = id_{TA}^*$  for all  $A \in \mathcal{K}$  and  $f : A \rightarrow B$ .

Conversely, a monad  $(T, \eta, \mu)$  defines the Kleisli triple  $(T, \eta, \_)^*$  with  $f^* = \mu_B \circ Tf$  for all  $f : A \rightarrow TB$ .

Haskell implements  $\_^*$  by the **bind operator**

$$\text{bind} = (>>) : TA \rightarrow (A \rightarrow TB) \rightarrow TB :$$

$$\text{bind}(t)(f) =_{\text{def}} f^*(t).$$

In Haskell,  $\eta$  and  $\mu$  are called *return* and *join*, respectively (see [here](#)).

## The Kleisli composition

$$\circ_T : (B \rightarrow TC) \times (A \rightarrow TB) \rightarrow (A \rightarrow TC)$$

combines *bind* with function composition:  $g \circ_T f =_{\text{def}} g^* \circ f$ .

$\eta$  and  $\circ_T$  induce the **Kleisli category over  $\mathcal{K}$** ,  $\mathcal{K}_T$ , whose objects are the objects of  $\mathcal{K}$  and whose morphisms from  $A$  to  $B$  are the  $\mathcal{K}$ -morphisms from  $A$  to  $TB$ . Composition is the Kleisli composition  $\circ_T$  and for all  $A \in \mathcal{K}$ , the  $\mathcal{K}_T$ -identity on  $A$  is defined as the unit instance  $\eta_A : A \rightarrow TA$ .

## Sample monads

Many functors defined in chapter 5 are monads. Unit, multiplication and bind are defined as follows:

- identity functor:  $\eta_A = \mu_A = id_A$ ,  $bind = \lambda a.\lambda f.f(a)$ .

- list functor:

$$\begin{array}{lll} \eta_A : A \rightarrow A^* & \mu_A : (A^*)^* \rightarrow A^* & bind : A^* \rightarrow (A \rightarrow B^*) \rightarrow B^* \\ a \mapsto (a) & (w_1, \dots, w_n) \mapsto w_1 \cdot \dots \cdot w_n & s \mapsto \lambda f. \mu_B(map(f)(s)) \end{array}$$

- powerset functor:

$$\begin{array}{ll} \eta_A : A \rightarrow \mathcal{P}(A) & \mu_A : \mathcal{P}(\mathcal{P}(A)) \rightarrow \mathcal{P}(A) \\ a \mapsto \{a\} & S \mapsto \bigcup S \end{array}$$

$$\begin{array}{ll} bind : \mathcal{P}(A) \rightarrow (A \rightarrow \mathcal{P}(B)) \rightarrow \mathcal{P}(B) & \\ S \rightarrow \lambda f. \bigcup \{f(s) \mid s \in S\} & \end{array}$$

$$\begin{array}{ll} \circ_{\mathcal{P}} : (B \rightarrow \mathcal{P}(C)) \times (A \rightarrow \mathcal{P}(B)) \rightarrow (A \rightarrow \mathcal{P}(C)) & \\ (g, f) \mapsto \lambda a. \bigcup \{g(b) \mid b \in f(a)\} & \end{array}$$

- finite-set functor  $\mathcal{P}_\omega$ : analogously
- $M$ -weighted-set functor: Let  $(M, +, 0, *, 1)$  be a semiring.

$$\begin{array}{ll} \eta_A : A \rightarrow M_\omega^A & \mu_A : M_\omega^{M_\omega^A} \rightarrow M_\omega^A \\ a \mapsto (\lambda x.0)[1/a] & g \mapsto \lambda a. \sum \{g(h) * h(a) \mid h \in \text{supp}(g)\} \\ \\ \text{bind} : M_\omega^A \rightarrow (A \rightarrow M_\omega^B) \rightarrow M_\omega^A & \\ h \mapsto \lambda f. \lambda b. \sum \{h(a) * f(a)(b) \mid a \in \text{supp}(g)\} & \end{array}$$

Kleisli composition is matrix multiplication:

$$\circ_{M_\omega^-} : (B \rightarrow M_\omega^C) \times (A \rightarrow M_\omega^B) \rightarrow (A \rightarrow M_\omega^C) \\ (g, f) \mapsto \lambda a. \sum \{g(b) * f(a) \mid b \in \text{supp}(g)\}$$

- distribution functor:

$$\begin{array}{ll} \eta_A : A \rightarrow \mathcal{D}_\omega(A) & \mu_A : \mathcal{D}_\omega(\mathcal{D}_\omega(A)) \rightarrow \mathcal{D}_\omega(A) \\ a \mapsto (\lambda x.0)[1/a] & g \mapsto \lambda a. \sum \{g(h) * h(a) \mid h \in \text{supp}(g)\} \end{array}$$

$$\begin{aligned}
 bind : \mathcal{D}_\omega(A) &\rightarrow (A \rightarrow \mathcal{D}_\omega(B)) \rightarrow \mathcal{D}_\omega(B) \\
 h &\mapsto \lambda f. \lambda b. \sum \{ h(a) * f(a)(b) \mid a \in supp(g) \}
 \end{aligned}$$

- exception functor:

$$\begin{array}{lll}
 \eta_A : A &\rightarrow A + X & \mu_A : (A + X) + X \rightarrow A + X \\
 a &\mapsto (a, 1) & ((a, 1), 1) \mapsto (a, 1) \\
 && ((x, 2), 1) \mapsto (x, 2) \\
 && (x, 2) \mapsto (x, 2)
 \end{array}$$

If  $X = 1$ , then Kleisli composition is composition of partial functions:

$$\begin{aligned}
 \circ_{\_+1} : (B \rightarrow C + 1) \times (A \rightarrow B + 1) &\rightarrow (A \rightarrow C + 1) \\
 g &\mapsto \lambda f. \lambda a. \begin{cases} (g(f(b)), 1) & \text{if } f(a) = (b, 1) \text{ for some } b \in B \\ (\epsilon, 2) & \text{if } f(a) = (\epsilon, 2) \end{cases}
 \end{aligned}$$

- reader functor:

$$\begin{array}{lll}
 \eta_A : A &\rightarrow A^X & \mu_A : (A^X)^X \rightarrow A^X & bind : A^X \rightarrow (A \rightarrow B^X) \rightarrow B^X \\
 a &\mapsto \lambda x. a & g &\mapsto \lambda x. g(x)(x) & h &\mapsto \lambda f. \lambda x. f(h(x))(x)
 \end{array}$$

- writer functor: Let  $(X, +, 0)$  be a monoid.

$$\begin{array}{lll} \eta_A : A \rightarrow A \times X & \mu_A : (A \times X) \times X \rightarrow A \times X \\ a \mapsto (a, 0) & ((a, x), y) \mapsto (a, x + y) \end{array}$$

$$\begin{array}{ll} bind : A \times X \rightarrow (A \rightarrow B \times X) \rightarrow B \times X \\ (a, x) \mapsto \lambda f. (\lambda(b, y). (b, x + y))(f(a)) \end{array}$$

- state functor:

$$\begin{array}{lll} \eta_A : A \rightarrow (A \times X)^X & \mu_A : ((A \times X)^X \times X)^X \rightarrow (A \times X)^X \\ a \mapsto \lambda x. (a, x) & g \mapsto \lambda(h, x). h(x) \circ g \\ bind : (A \times X)^X \rightarrow (A \rightarrow (B \times X)^X) \rightarrow (B \times X)^X \\ h \mapsto \lambda f. (\lambda(a, x). f(a)(x)) \circ h \end{array}$$

Let  $(T : Set \rightarrow Set, \eta, \mu)$  be a monad.

The **state monad transformer**

$$(T(\_) \times X)^X, \eta', \mu')$$

combines  $T$  with the state monad:

It maps a set  $A$  to the set  $T(A \times X)^X$  and a function  $f : A \rightarrow B$  to the function

$$\begin{aligned} T(f \times X)^X : T(A \times X)^X &\rightarrow T(B \times X)^X \\ g &\mapsto (\lambda(a, x). \eta(f(a), x)) \circ_T g \end{aligned}$$

$$\begin{aligned} \eta'_A : A &\rightarrow T(A \times X)^X & \mu'_A : (T(A \times X)^X \times X)^X &\rightarrow T(A \times X)^X \\ a &\mapsto \lambda x. \eta(a, x) & g &\mapsto (\lambda(h, x). h(x)) \circ_T g \end{aligned}$$

$$\begin{aligned} bind' : T(A \times X)^X &\rightarrow (A \rightarrow T(B \times X)^X) \rightarrow T(B \times X)^X \\ h &\mapsto \lambda f. (\lambda(a, x). f(a)(x)) \circ_T h \end{aligned}$$

Let  $adj = (L : \mathcal{K} \rightarrow \mathcal{L}, R : \mathcal{L} \rightarrow \mathcal{K}, \eta : Id_{\mathcal{K}} \rightarrow RL, \epsilon : LR \rightarrow Id_{\mathcal{K}})$  be an adjunction.

$M(adj) = (RL, \eta, R\epsilon L : RLRL \rightarrow RL)$  is a monad, called **the monad induced by  $adj$** .

## The monoid monad

Given the adjunction of section 19.2, the list functor  $\underline{\phantom{x}}^* : \mathbf{Set} \rightarrow \mathbf{Set}$  coincides with  $UMF$  and thus  $(\underline{\phantom{x}}^*, \eta, U\epsilon MF)$  is the monad induced by this adjunction.

## The sequence monad

Given the adjunction of section 19.3, the writer functor  $X^* \times \underline{\phantom{x}} : \mathbf{Set} \rightarrow \mathbf{Set}$  coincides with  $USF_X$  and thus  $(X^* \times \underline{\phantom{x}}, \eta, U\epsilon SF_X)$  is the monad induced by this adjunction.

## Term monads

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature.

The monad induced by the adjunction  $\text{adj}_\Sigma = (T_\Sigma, U_S, \eta, \epsilon)$  is called the **monad freely generated by  $\Sigma$**  (see section 17.11: [Term functors](#)).

The categories  $\text{Alg}_{M(\text{adj}_\Sigma)}$  and  $\text{Alg}_\Sigma$  are isomorphic.

Let  $V, V' \in \text{Set}_b^S$  (see chapter 7) and  $g \in T_\Sigma(V')^V$ .

The equation  $\text{bind}(t)(g) = g^*(t)$  (see above) yields the following definition of

$$\text{bind} : T_\Sigma(V) \rightarrow (V \rightarrow T_\Sigma(V')) \rightarrow T_\Sigma(V').$$

- For all  $s \in S$  and  $x \in V_s$ ,  $\text{bind}(x)(g) = g(x)$ .
- For all  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(V)_e$ ,  $\text{bind}(c(t))(g) = c^A(\text{bind}(t)(g))$ .
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in T_\Sigma(V)_{e_i}$ ,

$$\text{bind}(i(t))(g) = i(\text{bind}(t)(g)).$$

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t = ()\{i \rightarrow t_i \mid i \in I\} \in T_\Sigma(V)_e$ ,

$$\text{bind}(t)(g) = ()\{i \rightarrow \text{bind}(t_i)(g) \mid i \in I\}.$$

Hence for all  $t \in T_\Sigma(V)$  and  $g : V \rightarrow T_\Sigma(V')$ ,  $\text{bind}(t)(g) \in T_\Sigma(V')$  is obtained from  $t$  by replacing each leaf of  $t$  labelled with a variable  $x$  by  $g(x)$ .

Let  $T = U_S T_\Sigma$  and  $V \in Set_b^S$ .

The equation  $\mu_V = id_{TV}^*$  (see above) yields the following definition of the multiplication  $\mu : TT \rightarrow T$ :

- For all  $s \in S$  and  $t \in T_\Sigma(V)_s$ ,  $\mu_V(t) = t$ .
- For all  $c : e \rightarrow s \in C$  and  $t \in T_\Sigma(T_\Sigma(V))_e$ ,

$$\mu_V(c(t)) = c(\mu_V(t)).$$

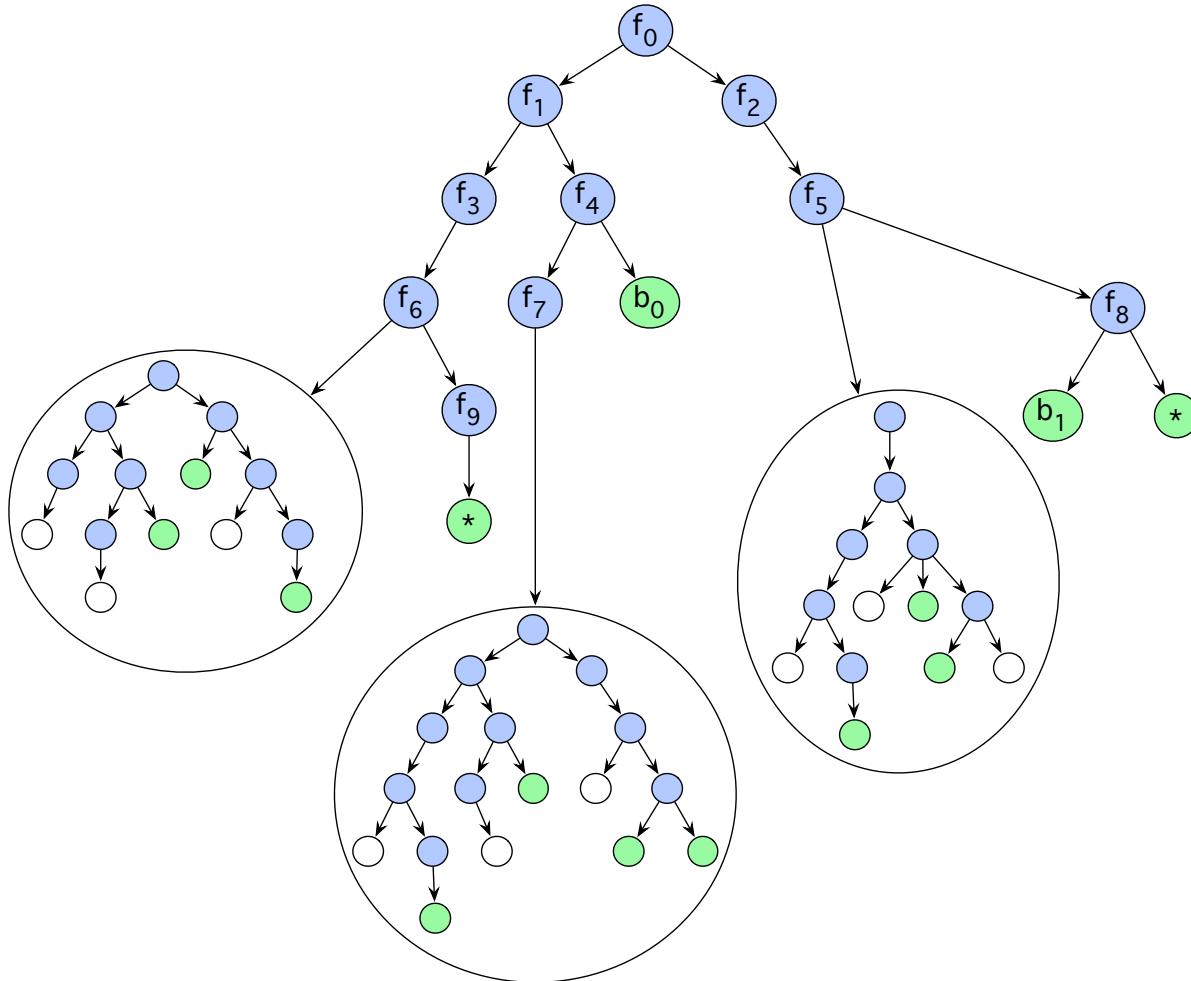
- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in T_\Sigma(T_\Sigma(V))_{e_i}$ ,

$$\mu_V(i(t)) = i(\mu_V(t)).$$

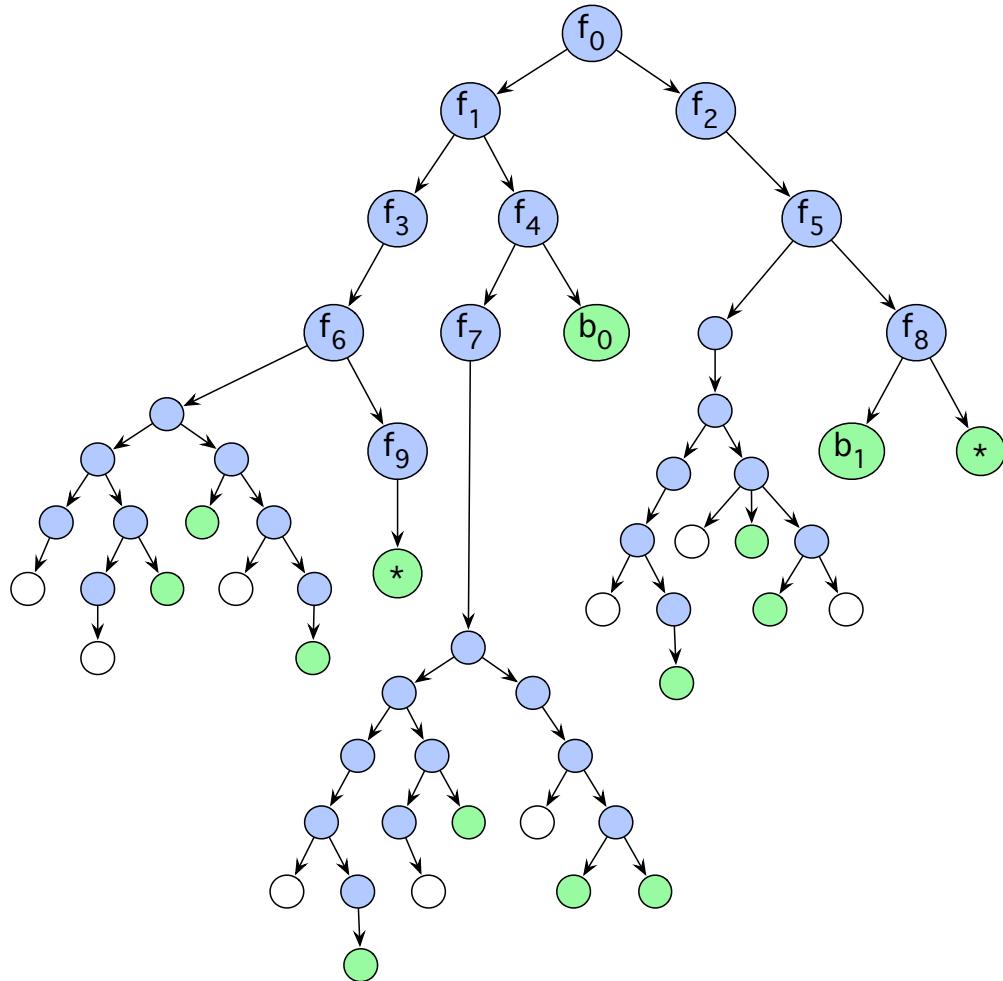
- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t = ()\{i \rightarrow t_i \mid i \in I\} \in T_\Sigma(T_\Sigma(V))_e$ ,

$$\mu_V(t) = ()\{i \rightarrow \mu_V(t_i) \mid i \in I\}.$$

Hence for all  $t \in T_\Sigma(T_\Sigma(V))$ ,  $\mu_V(t) \in T_\Sigma(V)$  is obtained from  $t$  by replacing the leaves of  $t$  with their labels.



A term  $t$  over  $T_\Sigma(V)$



The term over  $V$  that results from applying  $\mu_V : T_\Sigma(T_\Sigma(V)) \rightarrow T_\Sigma(V)$  to  $t$

Let  $M = (T : \mathcal{K} \rightarrow \mathcal{K}, \eta, \mu)$  be a monad.

An  $M$ -algebra or **Eilenberg-Moore algebra** is a  $T$ -algebra  $\alpha : TA \rightarrow A$  such that the following diagrams commute:

$$\begin{array}{ccc} A & \xrightarrow{\eta_A} & TA \\ id_A \searrow & & \downarrow \alpha \\ & A & \end{array} \quad \begin{array}{ccc} TTA & \xrightarrow{T\alpha} & TA \\ \mu_A \downarrow & & \downarrow \alpha \\ TA & \xrightarrow{\alpha} & A \end{array}$$

The category of  $M$ -algebras is denoted by  $\text{Alg}_M$ .  $\text{Alg}_M$  is a full subcategory of  $\text{Alg}_T$ .

The forgetful functor  $U : \text{Alg}_M \rightarrow \mathcal{K}$  has a left adjoint  $L : \mathcal{K} \rightarrow \text{Alg}_M$ .

Let  $adj_M = (L, U, \eta, \epsilon)$  be the corresponding adjunction.

The monad induced by  $adj_M$  coincides with  $M$ :  $M(adj_M) = M$ .

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive polynomial signature and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ . Then  $id_A^* : T_\Sigma(A) \rightarrow A$  is an Eilenberg-Moore  $T_\Sigma$ -algebra.

A **comonad** in  $\mathcal{K}$  is a triple  $CM = (D, \epsilon, \delta)$  consisting of a functor  $D : \mathcal{K} \rightarrow \mathcal{K}$  and natural transformations  $\epsilon : D \rightarrow Id_{\mathcal{K}}$  (**co-unit**) and  $\delta : D \rightarrow DD$  (**comultiplication**) such that the following diagrams commute:

The left diagram shows the multiplication law. It consists of three nodes:  $D$ ,  $DD$ , and  $D$ . There are two horizontal arrows: one from  $D$  to  $DD$  labeled  $\epsilon D$ , and another from  $DD$  to  $D$  labeled  $D\epsilon$ . There are also two diagonal arrows: one from  $D$  to  $D$  labeled  $id_D$ , and another from  $DD$  to  $D$  labeled  $id_D$ . A vertical arrow  $\delta$  goes from  $D$  up to  $DD$ . The right diagram shows the coassociativity law. It consists of four nodes:  $DDD$ ,  $DD$ ,  $DD$ , and  $D$ . There are three horizontal arrows: one from  $DDD$  to  $DD$  labeled  $\delta D$ , one from  $DD$  to  $DD$  labeled  $D\delta$ , and one from  $DD$  to  $D$  labeled  $\delta$ . There is also a vertical arrow  $\delta$  going from  $DD$  down to  $D$ .

A **co-Kleisli triple**  $(D, \epsilon, \underline{\phantom{x}}^\#)$  consists of a function  $D : \mathcal{K} \rightarrow \mathcal{K}$ , sets  $\epsilon = \{\epsilon_A : DA \rightarrow A \mid A \in \mathcal{K}\}$  and  $\underline{\phantom{x}}^\# = \{f^\# : DA \rightarrow DB \mid A, B \in \mathcal{K}, f : DA \rightarrow B\}$  such that for all  $A \in \mathcal{K}$ ,  $f : DA \rightarrow B$  and  $g : DB \rightarrow C$ ,

$$\epsilon_A^\# = id_{DA}, \quad \epsilon_B \circ f^\# = f, \quad (g \circ f^\#)^\# = g^\# \circ f^\#.$$

A co-Kleisli triple  $(D, \epsilon, \underline{\phantom{x}}^\#)$  defines the comonad  $(D, \epsilon, \delta)$  with  $Df = (f \circ \epsilon_A)^\#$  and  $\delta_A = id_{DA}^\#$  for all  $A \in \mathcal{K}$  and  $f : A \rightarrow B$ .

Conversely, a comonad  $(D, \epsilon, \delta)$  defines the co-Kleisli triple  $(D, \epsilon, \_)^\#$  with  $f^\# = Df \circ \delta_A$  for all  $f : DA \rightarrow B$ .

Haskell implements  $\_^\#$  by the **cobind operator**

$$\text{cobind} = (\lll) : (DA \rightarrow B) \rightarrow (DA \rightarrow DB) :$$

$$\text{cobind}(f)(d) =_{\text{def}} f^\#(d).$$

In Haskell,  $\epsilon$  and  $\delta$  are called *retract* and *duplicate*, respectively (see [here](#)).

The **co-Kleisli composition**

$$\circ^D = (=>) : (DB \rightarrow C) \times (DA \rightarrow B) \rightarrow (DA \rightarrow C)$$

combines *cobind* with function composition:  $g \circ^D f =_{\text{def}} g \circ f^\#$ .

$\epsilon$  and  $\circ^D$  induce the **co-Kleisli category over  $\mathcal{K}$** ,  $\mathcal{K}^D$ , whose objects are the objects of  $\mathcal{K}$  and whose morphisms from  $A$  to  $B$  are the  $\mathcal{K}$ -morphisms from  $DA$  to  $B$ . Composition is the co-Kleisli composition  $\circ^D$  and for all  $A \in \mathcal{K}$ , the  $\mathcal{K}^D$ -identity on  $A$  is defined as the co-unit instance  $\epsilon_A : DA \rightarrow A$ .

## Sample comonads

Many functors defined in chapter 5 are also comonads. Co-unit, comultiplication and cobind are defined as follows:

- identity functor:  $\epsilon_A = \delta_A = id_A$ .  $cobind = id_{A \rightarrow B}$ .

- list functor:

$$\begin{array}{ll} \epsilon_A : A^+ \rightarrow A & \delta_A : A^+ \rightarrow (A^+)^+ \\ (a_1, \dots, a_n) \mapsto a_1 & (a_1, \dots, a_n) \mapsto ((a_1, \dots, a_n), (a_2, \dots, a_n), \dots, a_n) \end{array}$$

$$\begin{array}{l} cobind : (A^+ \rightarrow B) \rightarrow (A^+ \rightarrow B^+) \\ f \mapsto \lambda(a_1, \dots, a_n). (f(a_1, \dots, a_n), f(a_2, \dots, a_n), \dots, f(a_n)) \end{array}$$

- reader functor: Let  $(X, +, 0)$  be a monoid.

$$\begin{array}{ll} \epsilon_A : A^X \rightarrow A & \delta_A : A^X \rightarrow (A^X)^X \\ h \mapsto h(0) & h \mapsto \lambda x. \lambda y. h(x + y) \end{array}$$

$$\begin{array}{l} cobind : (A^X \rightarrow B) \rightarrow (A^X \rightarrow B^X) \\ f \mapsto \lambda h. \lambda x. f(\lambda y. h(x + y)) \end{array}$$

- writer functor:

$$\begin{array}{ll} \epsilon_A : A \times X \rightarrow A & \delta_A : A \times X \rightarrow (A \times X) \times X \\ (a, x) \mapsto a & (a, x) \mapsto ((a, x), x) \end{array}$$

$$\begin{array}{ll} cobind : (A \times X \rightarrow B) \rightarrow (A \times X \rightarrow B \times X) & \\ f \mapsto \lambda(a, x). (f(a, x), x) & \end{array}$$

- costate functor:

$$\begin{array}{ll} \epsilon_A : A^X \times X \rightarrow A & \delta_A : A^X \times X \rightarrow (A^X \times X)^X \times X \\ (h, x) \mapsto h(x) & (h, x) \mapsto (\lambda y. (h, y), x) \end{array}$$

$$\begin{array}{ll} cobind : (A^X \times X \rightarrow B) \rightarrow (A^X \times X \rightarrow B^X \times X) & \\ f \mapsto \lambda(h, x). (\lambda y. f(h, y), x) & \end{array}$$

- labelled-tree functor:

$$\begin{array}{ll} \epsilon_A : ltr(X, A) \rightarrow A & \delta_A : ltr(X, A) \rightarrow ltr(X, ltr(X, A)) \\ t \mapsto t(\epsilon) & t \mapsto \lambda v. \lambda w. t(vw) \end{array}$$

$$\begin{array}{ll} cobind : (ltr(X, A) \rightarrow B) \rightarrow (ltr(X, A) \rightarrow ltr(X, B)) & \\ f \mapsto \lambda t. \lambda v. f(\lambda w. t(vw)) & \end{array}$$

- pointed-tree functor:

$$\begin{aligned}\epsilon_A : ltr(X, A) \times X^* &\rightarrow A \\ (t, w) &\mapsto t(w)\end{aligned}$$

$$\begin{aligned}\delta_A : ltr(X, A) \times X^* &\rightarrow ltr(X, ltr(X, A) \times X^*) \times X^* \\ (t, v) &\mapsto (\lambda w.(t, w), v)\end{aligned}$$

$$\begin{aligned}cobind : (ltr(X, A) \times X^* \rightarrow B) &\rightarrow (ltr(X, A) \times X^* \rightarrow ltr(X, B) \times X^*) \\ f &\mapsto \lambda(t, v).(\lambda w.f(t, w), v)\end{aligned}$$

Let  $adj = (L : \mathcal{K} \rightarrow \mathcal{L}, R : \mathcal{L} \rightarrow \mathcal{K}, \eta : Id_{\mathcal{K}} \rightarrow RL, \epsilon : LR \rightarrow Id_{\mathcal{K}})$  be an adjunction.

$CM(adj) = (\textcolor{red}{LR}, \epsilon, L\eta R : LR \rightarrow LRLR)$  is a comonad, called **the comonad induced by  $adj$** .

## The behavior comonad

Let  $X$  be a set. Given the adjunction of section 19.4, the reader functor  $\underline{\phantom{x}}^{X^*} : Set \rightarrow Set$  coincides with  $UBF_X$  and thus  $(\underline{\phantom{x}}^{X^*}, \epsilon, U\eta BF_X)$  is the comonad induced by this adjunction.

## Coterm comonads

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature and  $B = \bigcup \mathcal{I}$ .

The comonad induced by the adjunction  $\text{adj}_\Sigma = (U_S, DT_\Sigma, \eta, \epsilon)$  is called the **comonad cofreely generated by  $\Sigma$**  (see section 17.13: [Coterm functor](#)).

The categories  $coAlg_{CM(\text{adj}_\Sigma)}$  and  $coAlg_\Sigma$  are isomorphic.

Let  $C, C'$  be as in [Coterm functors](#) and  $g \in C'^{DT_\Sigma(C)}$ .

The equation  $\text{cobind}(g)(t) = g^\#(t)$  (see above) yields the following definition of

$$\text{cobind} : (DT_\Sigma(C) \rightarrow C') \rightarrow DT_\Sigma(C) \rightarrow DT_\Sigma(C') :$$

- For all  $s \in S$  and  $t \in DT_\Sigma(C)_s$ ,  $\text{cobind}(g)(t)(\epsilon) = g(t)$ .
- For all  $d : s \rightarrow e \in D$ ,  $t = c\{d \rightarrow t_d \mid d : s \rightarrow e \in D\} \in DT_\Sigma(C)_s$  and  $w \in (D \cup B)^*$ ,

$$\text{cobind}(g)(t)(dw) = \text{cobind}(g)(d^A(t))(w).$$

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $a \in DT_\Sigma(C)_{e_i}$ ,

$$\text{cobind}(g)(i(t)) = i(\text{cobind}(g)(t)).$$

- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t = ()\{i \rightarrow t_i \mid i \in I\} \in DT_\Sigma(C)_e$ ,

$$\text{cobind}(g)(t) = ()\{i \rightarrow \text{cobind}(g)(t_i) \mid i \in I\}.$$

Hence for all  $g : DT_{\Sigma}(C) \rightarrow C'$  and  $t \in DT_{\Sigma}(C)$ ,  $cobind(g)(t) \in DT_{\Sigma}(C')$  is obtained from  $t$  by replacing  $t(w)$  with the  $g$ -image of the subtree of  $t$  with root position  $w$ , i.e.,

$$cobind(g)(t)(w) = g(\lambda v.t(wv)).$$

Let  $D = U_S DT_{\Sigma}$  and  $C \in Set_b^S$ .

The equation  $\delta_C = id_{DC}^{\#}$  (see above) yields the following definition of the comultiplication  $\delta : D \rightarrow DD$ :

- For all  $s \in S$  and  $t \in DT_{\Sigma}(C)_s$ ,  $\delta_C(t)(\epsilon) = t$ .
- For all  $d : s \rightarrow e \in D$ ,  $t = c\{d \rightarrow t_d \mid d : s \rightarrow e \in D\} \in DT_{\Sigma}(C)_s$  and  $w \in (D \cup B)^*$ ,

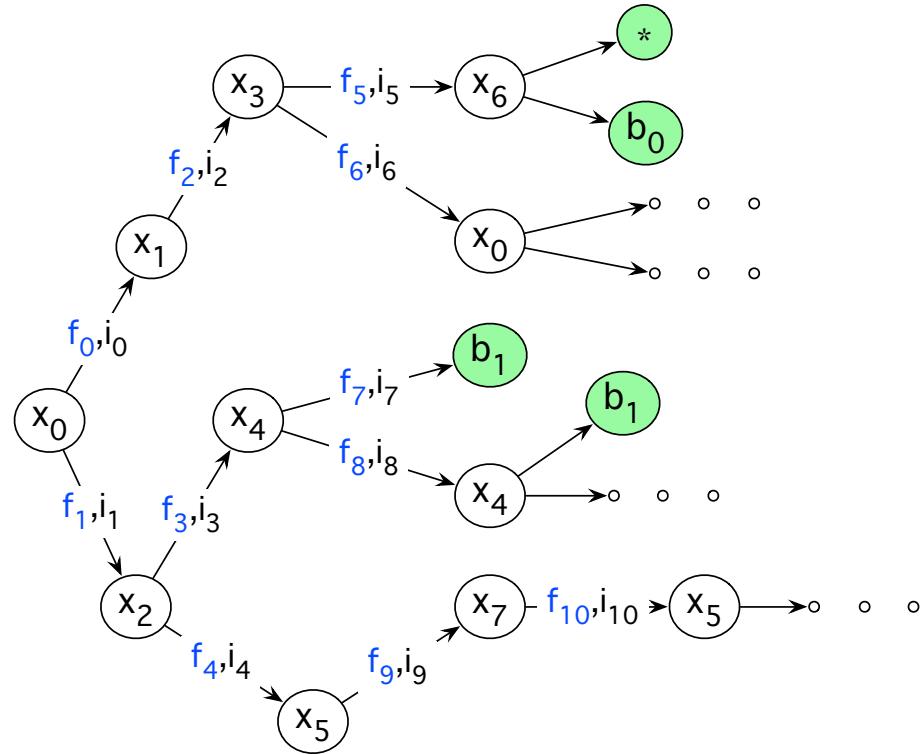
$$\delta_C(t)(dw) = \delta_C(t_d)(w).$$

- For all sum types  $e = \coprod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$ ,  $i \in I$  and  $t \in DT_{\Sigma}(C)_{e_i}$ ,  $\delta_C(i(t)) = i(\delta_C(t))$ .
- For all product types  $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S, \mathcal{I})$  and  $t = ()\{i \rightarrow t_i \mid i \in I\} \in DT_{\Sigma}(C)_e$ ,

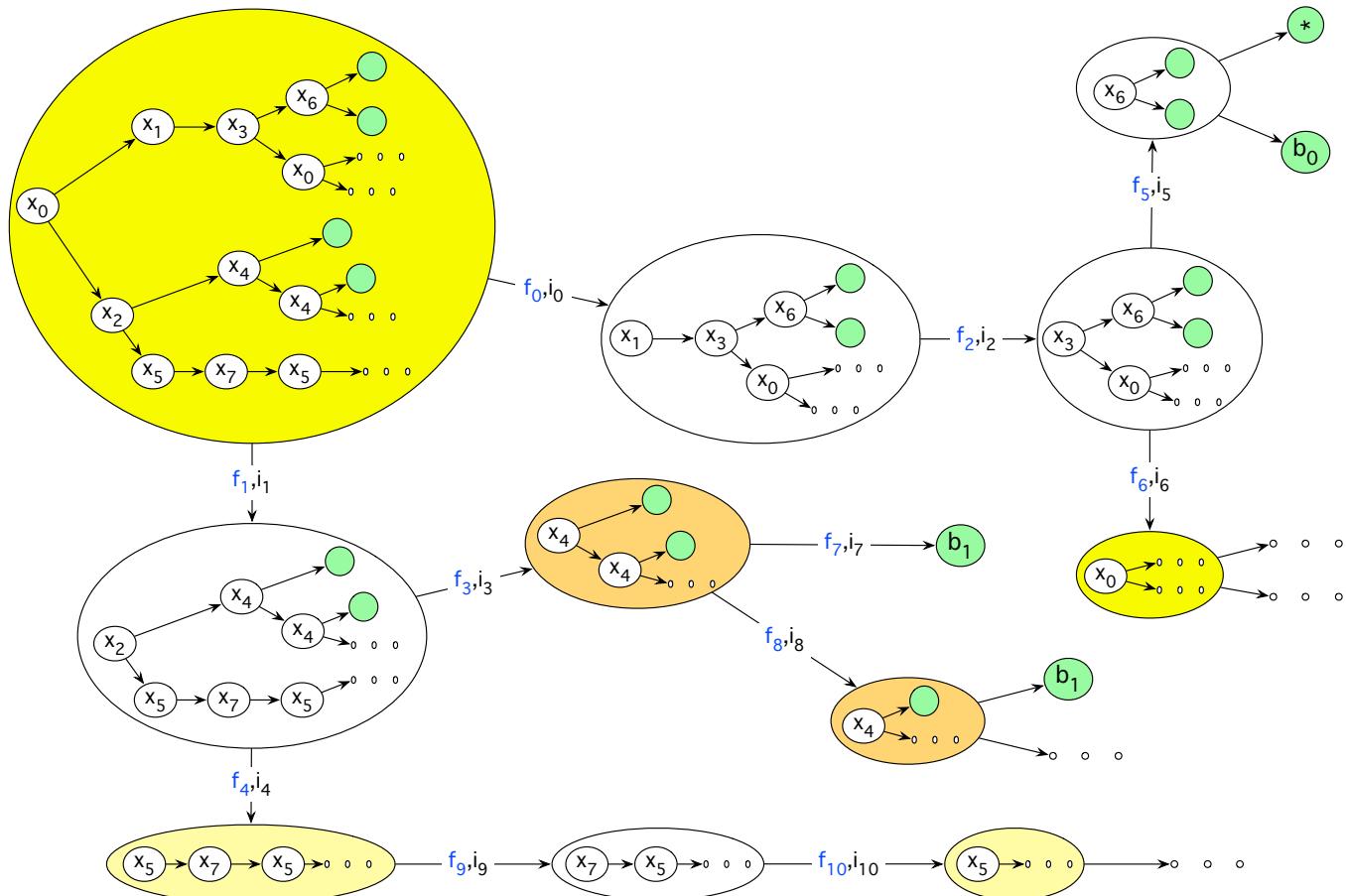
$$\delta_C(t) = ()\{i \rightarrow \delta_C(t_i) \mid i \in I\}.$$

By (9) in chapter 9, for all  $t \in DT_{\Sigma}(C)$ ,  $\delta_C(t) \in DT_{\Sigma}(DT_{\Sigma}(C))$  is obtained from  $t$  by replacing  $t(w)$  with the subtree of  $t$  with root position  $w$ , i.e.,

$$\delta_C(t)(w) = \lambda v.t(wv).$$



A cocomonad  $t$  over  $C$



*The coterm over  $DT_{\Sigma}(C)$  that results  
from applying  $\delta_V : DT_{\Sigma}(C) \rightarrow DT_{\Sigma}(DT_{\Sigma}(C))$  to  $t$*

Let  $CM = (D : \mathcal{K} \rightarrow \mathcal{K}, \epsilon, \delta)$  be a comonad.

A  **$CM$ -coalgebra** or **Eilenberg-Moore coalgebra** is a  $D$ -coalgebra  $\beta : A \rightarrow DA$  such that the following diagrams commute:

$$\begin{array}{ccc} A & \xleftarrow{\epsilon_A} & DA \\ id_A \swarrow & & \uparrow \beta \\ & A & \end{array} \quad \begin{array}{ccc} DDA & \xleftarrow{D\beta} & DA \\ \delta_A \uparrow & & \uparrow \beta \\ DA & \xleftarrow{\beta} & A \end{array}$$

The category of  $CM$ -coalgebras is denoted by  $coAlg_{CM}$ .  $coAlg_{CM}$  is a full subcategory of  $coAlg_D$ .

The forgetful functor  $U : coAlg_{CM} \rightarrow \mathcal{K}$  has a right adjoint  $R : \mathcal{K} \rightarrow coAlg_{CM}$ .

Let  $adj_{CM} = (U, R, \eta, \epsilon)$  be the corresponding adjunction.

The comonad induced by  $adj_{CM}$  coincides with  $CM$ :  $CM(adj_{CM}) = CM$ .

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive polynomial signature and  $\mathcal{A}$  be a  $\Sigma$ -algebra with carrier  $A$ . Then  $id_A^\# : A \rightarrow DT_\Sigma(A)$  is an Eilenberg-Moore  $DT_\Sigma$ -coalgebra.

Given a monad  $M = (T, \eta, \mu)$ , a distributive law  $\lambda : TD \rightarrow DT$  is  **$M$ -compatible** if the following diagrams commute:

$$\begin{array}{ccc} D & \xrightarrow{\eta D} & TD \\ & \searrow D\eta & \downarrow \lambda \\ & & DT \end{array}$$

$$\begin{array}{ccccc} TTD & \xrightarrow{T\lambda} & TDT & \xrightarrow{\lambda T} & DTT \\ \mu D \downarrow & & & & D\mu \downarrow \\ TD & \xrightarrow{\lambda} & DT & & \end{array}$$

Given a comonad  $CM = (D, \epsilon, \delta)$ , a distributive law  $\lambda : TD \rightarrow DT$  is  **$CM$ -compatible** if the following diagrams commute:

$$\begin{array}{ccc} T & \xleftarrow{\epsilon T} & DT \\ \nwarrow T\epsilon & & \uparrow \lambda \\ TD & & \end{array}$$

$$\begin{array}{ccccc} DDT & \xleftarrow{D\lambda} & DTD & \xleftarrow{\lambda D} & TDD \\ \delta T \uparrow & & & & T\delta \uparrow \\ DT & \xleftarrow{\lambda_A} & TD & & \end{array}$$

## Examples

Given a monad  $M = (T, \eta, \mu)$  in  $\text{Set}$ , the strength  $st^{T,A}$  of  $T$  and  $A$  is  $M$ -compatible.

Given a monoid  $A$  with multiplication  $\cdot$  and unit  $e$ ,

$$CM = ((-)^A, \epsilon, \delta)$$

with  $\epsilon_B(f) = f(e)$  and  $\delta_B(f) = \lambda a. \lambda b. f(a \cdot b)$  for all sets  $B$  and  $f \in B^A$  is a comonad and  $st^{T,A}$  is  $CM$ -compatible.

Given a  $T$ -algebra  $\alpha : TB \rightarrow B$ , let  $D = (-)^A \times B$ .

$$\lambda : TD \rightarrow DT$$

with

$$\lambda_X : TDX = T(X^A \times B) \xrightarrow{\langle T(\pi_1), T(\pi_2) \rangle} T(X^A) \times TB \xrightarrow{st_X^{T,A} \times \alpha} (TX)^A \times B = DTX$$

is an  $M$ -compatible distributive law. □

## Recursive functions defined by adjunctions or distributive laws

**Theorem RECFUN4** (inspired by [69])

Let  $\Sigma = (S, \mathcal{I}, C)$  be a constructive signature,

$$(L : Mod(S, \mathcal{I}) \rightarrow \mathcal{K}, R : \mathcal{K} \rightarrow Mod(S, \mathcal{I}), \eta : Id_{Mod(S, \mathcal{I})} \rightarrow RL, \epsilon : LR \rightarrow Id_{\mathcal{K}})$$

be an adjunction,  $\mathcal{A}$  be an initial  $\Sigma$ -algebra with carrier  $A$  and  $B \in \mathcal{K}$  such that  $R(B)$  is a  $\Sigma$ -algebra.

There is a unique  $\mathcal{K}$ -morphism  $d : L(A) \rightarrow B$  such that for all  $c : e \rightarrow s \in C$ ,

$$d_s^\# \circ c^{\mathcal{A}} = c^{R(B)} \circ d_e^\#. \quad (1)$$

$d$  is  **$(L, R)$ -recursive** if (1) holds true for all  $c \in C$ .

*Proof.* Let  $d = \epsilon_B \circ L(fold^{R(B)})$ . Since  $(L, R, \eta, \epsilon)$  is an adjunction, there is a unique  $S$ -sorted function  $d^\# : A \rightarrow R(B)$  such that  $\epsilon_B \circ L(d^\#) = d$ . Hence  $d^\# = fold^{R(B)}$  and thus  $d^\#$  is  $\Sigma$ -homomorphic, i.e., (1) holds true.

Let  $f, g : L(A) \rightarrow B$  be two  $\mathcal{K}$ -morphisms such that  $f^\#$  and  $g^\#$  are  $\Sigma$ -homomorphic. Since  $\mathcal{A}$  is initial in  $Alg_\Sigma$ ,  $f^\# = fold^{R(B)} = g^\#$ . Hence  $f = \epsilon_B \circ L(f^\#) = \epsilon_B \circ L(g^\#) = g$ .  $\square$

Like Theorem RECFUN1, Theorem RECFUN4 covers the simultaneous definition of the elements of a function tuple  $d = (f_i : A \rightarrow B_i)_{i \in I}$ . Provided that  $S$  is a singleton,

$$(L = \Delta^I : Set \rightarrow Set^I, R = \prod_{i \in I} : Set^I \rightarrow Set)$$

is the appropriate adjunction (see section 19.9). Since  $d^\# = \langle f_i \rangle_{i \in I}$ , RECFUN4 (1) can be transformed into a formula of the form RECFUN1 (1).

In the following example, Theorem RECFUN4 is applied to the reader-writer adjunction (see section 19.8) in order to define a *binary* function inductively.

## Example

Let  $\Sigma = Nat$ ,  $L = \underline{\phantom{x}} \times \mathbb{N}$  and  $R = \underline{\phantom{x}}^{\mathbb{N}}$ . Then  $L(\mathbb{N}) = \mathbb{N}^2$   $R(\mathbb{N}) = \mathbb{N}^{\mathbb{N}}$  and we interpret the arrows  $zero : 1 \rightarrow nat$  and  $succ : nat \rightarrow nat$  on  $R(\mathbb{N})$  as follows:  $zero^{R(\mathbb{N})} = \lambda n. n$  and  $succ^{R(\mathbb{N})} = \lambda f. \lambda n. f(n) + 1$ . The addition of natural numbers,  $add : L(\mathbb{N}) \rightarrow \mathbb{N}$ , satisfies the equations

$$add(0, n) = n, \tag{2}$$

$$add(m + 1, n) = add(m, n) + 1 \tag{3}$$

for all  $m, n \in \mathbb{N}$  iff  $add^\# = curry(add) : \mathbb{N} \rightarrow R(\mathbb{N})$  satisfies (1), i.e.,

$$add^\#(0) = zero^{R(\mathbb{N})}, \quad (4)$$

$$add^\#(m + 1) = succ^{R(\mathbb{N})}(add^\#(m)) \quad (5)$$

for all  $m \in \mathbb{N}$ .

*Proof.* “ $\Rightarrow$ ”: Let (2) and (3) hold true. Then

$$add^\#(0) = curry(add)(0) = (\lambda m. \lambda n. add(m, n))(0) = \lambda n. add(0, n) \stackrel{(2)}{=} \lambda n. n = zero^{R(\mathbb{N})},$$

and for all  $m \in \mathbb{N}$ ,

$$\begin{aligned} add^\#(m + 1) &= curry(add)(m + 1) = (\lambda m'. \lambda n. add(m', n))(m + 1) = \lambda n'. add(m + 1, n) \\ &\stackrel{(3)}{=} add(m, n) + 1 = d^\#(m)(n) + 1 = (\lambda f. \lambda n. f(n) + 1)(d^\#(m)) = succ^{R(\mathbb{N})}(d^\#(m)), \end{aligned}$$

i.e., (4) and (5) are valid.

“ $\Leftarrow$ ”: Let (4) and (5) hold true. Then (2) and (3) follow from a rearrangement of the preceding equations.

Hence by Theorem RECFUN4,  $add$  is uniquely defined by (2) and (3). □

**Exercise 19** Let  $\Sigma, L, R$  and the  $\Sigma$ -algebra  $R(\mathbb{N})$  be defined as in the previous example. Show the following equivalence:

The multiplication of natural numbers,  $mult : L(\mathbb{N}) \rightarrow \mathbb{N}$ , satisfies the equations

$$mult(0, n) = 0, \tag{6}$$

$$mult(m + 1, n) = mult(m, n) + n \tag{7}$$

for all  $m, n \in \mathbb{N}$  iff  $mult^\# = curry(mult) : \mathbb{N} \rightarrow R(\mathbb{N})$  satisfies (4) and (5) with  $mult$  instead of  $add$ .

Hence by Theorem RECFUN4,  $mult$  is uniquely defined by (6) and (7). □

It must be noted that the transformation of the equations for the binary functions  $add$  and  $mult$  into inductive definitions of their curried versions is simple because the equations exhibit an inductive definition only in the first argument. For binary functions where the inductive definition extends over several arguments, the transformation is more difficult and may lead to nested folds (see, e.g., the Ackermann function in [Sample inductively defined functions](#)).

## Theorem RECFUN5 (inspired by [69])

Let  $\Sigma = (S, \mathcal{I}, D)$  be a destructive signature,

$$(L : \mathcal{K} \rightarrow Mod(S, \mathcal{I}), R : Mod(S, \mathcal{I}) \rightarrow \mathcal{K}, \eta : Id_{\mathcal{K}} \rightarrow RL, \epsilon : LR \rightarrow Id_{Mod(S, \mathcal{I})})$$

be an adjunction,  $\mathcal{A}$  be a final  $\Sigma$ -algebra with carrier  $A$  and  $B \in \mathcal{K}$  such that  $L(B)$  is a  $\Sigma$ -algebra.

There is a unique  $\mathcal{K}$ -morphism  $c : B \rightarrow R(A)$  such that for all  $d : s \rightarrow e \in D$ ,

$$d^{\mathcal{A}} \circ c_s^* = c_e^* \circ d^{\mathcal{A}}. \quad (2)$$

$c$  is  **$(L, R)$ -corecursive** if (2) holds true for all  $d \in D$ .

*Proof.* Let  $c = R(unfold^{L(B)}) \circ \eta_B$ . Since  $(L, R, \eta, \epsilon)$  is an adjunction, there is a unique  $S$ -sorted function  $c^* : L(B) \rightarrow A$  such that  $R(c^*) \circ \eta_B = c$ . Hence  $c^* = unfold^{L(B)}$  and thus  $c^*$  is  $\Sigma$ -homomorphic.

Let  $f, g : B \rightarrow R(A)$  be two  $\mathcal{K}$ -morphisms such that  $f^*$  and  $g^*$  are  $\Sigma$ -homomorphic. Since  $\mathcal{A}$  is final in  $Alg_{\Sigma}$ ,  $f^* = unfold^{L(B)} = g^*$ . Hence  $f = R(f^*) \circ \eta_B = R(g^*) \circ \eta_B = g$ .

□

Like Theorem RECFUN2, Theorem RECFUN5 covers the simultaneous definition of the elements of a function tuple  $d = (f_i : B_i \rightarrow A)_{i \in I}$ . Provided that  $S$  is a singleton,

$$(L = \coprod_{i \in I} : Set^I \rightarrow Set, R = \Delta^I : Set \rightarrow Set^I, )$$

is the appropriate adjunction (see section 19.10). Since  $c^* = [f_i]_{i \in I}$ , RECFUN5 (1) can be transformed into a formula of the form RECFUN2 (1).

Given a constructive or destructive signature  $\Sigma$ , Lemma BINIFIN suggests to define operations on the initial or final  $\Sigma$ -algebra in terms of distributive laws of or over  $H_\Sigma$ , respectively:

## Theorem RECFUN6

Let  $\Sigma = (S, \mathcal{I}, F)$  be a constructive signature and  $\lambda : H_\Sigma D \rightarrow DH_\Sigma$  be a distributive law of  $H_\Sigma$  over some endofunctor  $D$  on  $Set^S$ .

Let  $\mu\Sigma$  be initial in  $Alg_\Sigma$ . Then  $\alpha : H_\Sigma(\mu\Sigma) \rightarrow \mu\Sigma$  with  $H_\Sigma(\mu\Sigma)_s = \coprod_{c:e \rightarrow s \in F} \mu\Sigma_e$  and  $\alpha_s = [c^{\mu\Sigma}]_{c:e \rightarrow s \in F}$  for all  $s \in S$  is initial in  $Alg_{H_\Sigma}$  (see  $\Sigma$ -functors).

An  $S$ -sorted function  $f : \mu\Sigma \rightarrow D(\mu\Sigma)$  is  **$\lambda$ -recursive** if  $(\alpha, f)$  is a  $\lambda$ -bialgebra, i.e.,

$$f \circ \alpha = D\alpha \circ \lambda_{\mu\Sigma} \circ H_\Sigma(f)$$

or, equivalently, for all  $c : e \rightarrow s \in F$ ,

$$f \circ c^{\mu\Sigma} = (D\alpha)_s \circ \lambda_{\mu\Sigma, s} \circ \iota_c \circ f_e.$$

There is a unique  $\lambda$ -recursive function  $f : \mu\Sigma \rightarrow D(\mu\Sigma)$ .

*Proof.* Lemma **BINIFIN** (3). □

## Theorem RECFUN7

Let  $\Sigma = (S, \mathcal{I}, F)$  be a destructive signature and  $\lambda : TH_\Sigma \rightarrow H_\Sigma T$  be a distributive law of some endofunctor  $T$  on  $Set^S$  over  $H_\Sigma$ .

Let  $\nu\Sigma$  be final in  $Alg_\Sigma$ . Then  $\beta : \nu\Sigma \rightarrow H_\Sigma(\nu\Sigma)$  with  $H_\Sigma(\nu\Sigma)_s = \prod_{d:s \rightarrow e \in F} \nu\Sigma_e$  and  $\beta_s = \langle d^{\nu\Sigma} \rangle_{d:s \rightarrow e \in F}$  for all  $s \in S$  is final in  $coAlg_{H_\Sigma}$  (see  **$\Sigma$ -functors**).

An  $S$ -sorted function  $f : T(\nu\Sigma) \rightarrow \nu\Sigma$  is  **$\lambda$ -corecursive** if  $(f, \beta)$  is a  $\lambda$ -bialgebra, i.e.,

$$\beta \circ f = H_\Sigma(f) \circ \lambda_{\nu\Sigma} \circ T\beta$$

or, equivalently, for all  $d : s \rightarrow e \in F$ ,

$$d^{\nu\Sigma} \circ f = f_e \circ \pi_d \circ \lambda_{\nu\Sigma,s} \circ (T\beta)_s.$$

There is a unique  $\lambda$ -corecursive function  $f : T(\nu\Sigma) \rightarrow \nu\Sigma$ .

*Proof.* Lemma BINIFIN (4). □

**Example** (sum of streams; see Stream calculus and [26])

Let  $X$  be a semiring. The functions  $\text{in} : X \rightarrow X^{\mathbb{N}}$  and  $+ : X^{\mathbb{N}} \times X^{\mathbb{N}} \rightarrow X^{\mathbb{N}}$  are defined as follows: Let  $\beta = \langle \text{head}, \text{tail} \rangle : X^{\mathbb{N}} \rightarrow H_{\text{Stream}(X)}(X^{\mathbb{N}}) = X \times X^{\mathbb{N}}$ .

Define  $T : \text{Set} \rightarrow \text{Set}$  and  $\lambda : TH_{\text{Stream}(X)} \rightarrow H_{\text{Stream}(X)}T$  as follows:

For all  $A \in \text{Set}$ ,  $h \in \text{Mor}(\text{Set})$  and  $((x, a), (y, b)) \in (X \times A) \times (X \times A) = TH_{\text{Stream}(X)}A$ ,

$$T(A) = A \times A, \quad T(h) = h \times h,$$

$$\lambda_A((x, a), (y, b)) = (x + y, (a, b)) \in X \times (A \times A) = H_{\text{Stream}(X)}TA.$$

A function  $+ : X^{\mathbb{N}} \times X^{\mathbb{N}} \rightarrow X^{\mathbb{N}}$  satisfies the equations

$$\text{head}(s + s') = \text{head}(s) + \text{head}(s') \tag{1}$$

$$\text{tail}(s + s') = \text{tail}(s) + \text{tail}(s') \tag{2}$$

iff  $+$  is a  $\lambda$ -corecursive, i.e., the following equations hold true:

$$\begin{aligned} \text{head} \circ + &= \pi_{\text{head}} \circ \lambda_{\nu\Sigma} \circ (\beta \times \beta) \\ \text{tail} \circ + &= + \circ \pi_{\text{tail}} \circ \lambda_{\nu\Sigma} \circ (\beta \times \beta) \end{aligned}$$

*Proof.* For all  $s, s' \in X^{\mathbb{N}}$ ,

$$\begin{aligned} (\text{head} \circ +)(s, s') &= \text{head}(s + s'), \\ (\pi_{\text{head}} \circ \lambda_{\nu\Sigma} \circ (\beta \times \beta))(s, s') &= \pi_{\text{head}}(\lambda_{\nu\Sigma}((\text{head}(s), \text{tail}(s)), (\text{head}(s'), \text{tail}(s')))) \\ &= \pi_{\text{head}}(\text{head}(s) + \text{head}(s'), (\text{tail}(s), \text{tail}(s'))) = \text{head}(s) + \text{head}(s'), \\ (\text{tail} \circ +)(s, s') &= \text{tail}(s + s'), \\ (+ \circ \pi_{\text{tail}} \circ \lambda_{\nu\Sigma} \circ (\beta \times \beta))(s, s') &= +(\pi_{\text{tail}}(\lambda_{\nu\Sigma}((\text{head}(s), \text{tail}(s)), (\text{head}(s'), \text{tail}(s'))))) \\ &= +(\pi_{\text{tail}}(\text{head}(s) + \text{head}(s'), (\text{tail}(s), \text{tail}(s')))) = \text{tail}(s) + \text{tail}(s'). \end{aligned}$$

Hence by **Theorem RECFUN7**, equations (1) and (2) have a unique solution  $+$ . □

## External examples

**Queues** (see Rothe et al. ([140], p. 185)

A specification of queues with entries from a set  $A$ :

**BEGIN Queue[ A : TYPE ] : CLASSSPEC**

**METHOD**

put : [Self, A]  $\rightarrow$  Self;

top : Self  $\rightarrow$  Lift[[A,Self]];

**CONSTRUCTOR**

new : Self;

**ASSERTION SELFVAR** x : Self

q\_empty : x.top  $\cong$   $\perp$  **IMPLIES**

FORALL(a : A) . x.put(a).top  $\cong$  up(a,x);

q\_filled :

FORALL(a1:A, y : Self) . x.top  $\cong$  up(a1,y) **IMPLIES**

FORALL(a2 : A) . x.put(a2).top  $\cong$  up(a1, y.put(a2));

**CREATION**

q\_new : new.top  $\cong$   $\perp$ ;

**END Queue**

**Signature**  $Queue = (S, \mathcal{I}, F)$ :

$$\begin{aligned} S &= \{queue\}, \quad F = \{ \quad new : queue, \\ &\quad top : queue \rightarrow 1 + (A \times queue), \\ &\quad put : queue \times A \rightarrow queue \} \end{aligned}$$

**Axioms** for  $Queue$ :

$$top(new) = \epsilon$$

$$\forall q, x, a : (top(q) = (x, 1) \Rightarrow top(put(q, a)) = ((a, q), 2))$$

$$\forall q, q', a, a' : (top(q) = ((a, q'), 2) \Rightarrow top(put(q, a')) = ((a, put(q', a')), 2)))$$

The **model**  $M$  of  $Queue$  (following [140], p. 181):

$$\begin{aligned} M_{queue} &= (\mathbb{N} \cup \{\omega\}) \times (\mathbb{N} \rightarrow A), \\ new^M &= (0, \lambda i. a) \quad \text{for some } a \in A. \end{aligned}$$

For all  $(n, f) \in M_{queue}$  and  $a \in A$ ,

$$\begin{aligned} top^M(n, f) &= \begin{cases} (\epsilon, 1) & \text{if } n = 0, \\ ((f(0), \lambda i. f(i + 1)), 2) & \text{otherwise,} \end{cases} \\ put^M((n, f), a) &= \begin{cases} (n, f) & \text{if } n = \omega, \\ (n + 1, \lambda i. \text{if } i = n \text{ then } a \text{ else } f(i)) & \text{otherwise.} \end{cases} \end{aligned}$$

Hutton's motto:

$$\begin{array}{lll} \text{denotational semantics} & = & \text{folding of syntax trees} \\ \text{operational semantics} & = & \text{unfolding to transition trees} \end{array} \quad \begin{array}{l} = \text{ evaluation in an algebra} \\ = \text{ execution in a coalgebra} \end{array}$$

## Arithmetic expressions ([74], sections 2-4)

Let  $B$  be a set,  $\Sigma = (S, \mathcal{I}, F)$  with

$$S = \{exp\}, \quad F = \{add : exp \times exp \rightarrow exp\},$$

$\Sigma(B) = (S, \mathcal{I}, F)$  with

$$\begin{aligned} S &= \{exp\}, \quad F = \{ & val : B \rightarrow exp, \\ & add : exp \times exp \rightarrow exp \}. \end{aligned}$$

$\Sigma(B, V) = (S, \mathcal{I}, F)$  with

$$\begin{aligned} S &= \{exp\}, \quad F = \{ & val : B \rightarrow exp, \\ & var : V \rightarrow exp, \\ & add : exp \times exp \rightarrow exp \}. \end{aligned}$$

Hence for all sets and functions  $X$ ,

$$\begin{aligned} H_{\Sigma}(X) &= X \times X, \\ H_{\Sigma(B)}(X) &= B + (X \times X), \\ H_{\Sigma(B,V)}(X) &= B + V + (X \times X) \end{aligned}$$

(see  $\Sigma$ -functors).

Let  $A$  be a  $\Sigma(B)$ -algebra.

Then  $[val^A, add^A] : H_{\Sigma(B)}(A) \rightarrow A$  is the corresponding  $H_{\Sigma(B)}$ -algebra.

Let  $A$  be a  $\Sigma$ -algebra and  $T_{\Sigma}(B)$  be the set of all finite trees whose inner nodes are labelled with  $F$  and whose leaves are labelled with  $B$ .

$T_{\Sigma}(B)$  is the free  $\Sigma$ -algebra over  $B$ :

$$\begin{array}{ccc} B & \xrightarrow{\text{inc}} & T_{\Sigma}(B) \cong \text{Expr}(B) \quad (\text{see [74], p.2}) \\ f \searrow & = & \nearrow f^* \\ & A_{exp} & \end{array}$$

For all  $t, t' \in T_\Sigma(B)$ ,  $\text{add}^{T_\Sigma(B)}(t, t') = \text{add}(t, t')$ .

For all  $t, t' \in T_\Sigma(B)$  and  $b \in B$ ,

$$f^*(\text{val}(b)) = f(b) \quad \text{and} \quad f^*(\text{add}(t, t')) = \text{add}^A(f^*(t), f^*(t')).$$

$T_\Sigma(B)$  is also a initial  $\Sigma(B)$ -algebra:

For all  $b \in B$ ,  $\text{val}^{T_\Sigma(B)}(b) = \text{val}(b)$ .

Let  $A$  be a  $\Sigma(B)$ -algebra.

The unique  $\Sigma(B)$ -homomorphism  $\text{fold}^A : T_\Sigma(B) \rightarrow A$  is defined as follows: For all  $b \in B$  and  $t, t' \in T_\Sigma(B)$ ,

$$\text{fold}^A(\text{val}(b)) = \text{val}^A(b) \quad \text{and} \quad \text{fold}^A(\text{add}(t, t')) = \text{add}^A(\text{fold}^A(t), \text{fold}^A(t')).$$

$\text{fold}^A = \text{deno}$  (see [74], p.2) for  $f = \text{val}^A$  and  $g = \text{add}^A$ .

The functor  $\text{T}_\Sigma = \text{Expr} : \text{Set} \rightarrow \text{Alg}_\Sigma$  is left adjoint to the forgetful functor from  $\text{Alg}_\Sigma$  to  $\text{Set}$ . For all functions  $f : A \rightarrow C$ ,  $\text{T}_\Sigma(f) = (\text{inc} \circ f)^* : T_\Sigma(A) \rightarrow T_\Sigma(C)$ .

Let  $A$  be a  $\Sigma(B)$ -algebra.

$$\begin{array}{ccc}
 H_{\Sigma(B)}(T_{\Sigma}(B)) & \xrightarrow{[val^{T_{\Sigma}(B)}, add^{T_{\Sigma}(B)}]} & T_{\Sigma}(B) \\
 H_{\Sigma(B)}(fold^A) \downarrow & & \downarrow fold^A \\
 H_{\Sigma(B)}(A) & \xrightarrow{[val^A, add^A]} & A
 \end{array}$$

The  $\Sigma(\mathbb{Z})$ -algebra  $\mathbb{Z}$  of integers:  $val^{\mathbb{Z}} = id_{\mathbb{Z}}$ ,  $add^{\mathbb{Z}} = (+)$

$$fold^{\mathbb{Z}} = id_{\mathbb{Z}}^* = \text{eval} : T_{\Sigma}(\mathbb{Z}) \rightarrow \mathbb{Z} \text{ (see [74], p.2)}$$

The  $\Sigma(\mathbb{Z}, V)$ -algebra  $\mathbb{Z}^V$  of integer stores:

$$\begin{aligned}
 val^{\mathbb{Z}^V} &= \lambda n. \lambda s. n, \\
 var^{\mathbb{Z}^V} &= \lambda x. \lambda s. s(x), \\
 add^{\mathbb{Z}^V} &= \lambda(f, g). \lambda s. f(s) + g(s).
 \end{aligned}$$

The  $\Sigma(\mathbb{Z}, V)$ -algebra  $Com^*$  of assembler programs:

$$\begin{aligned}
 val^{Com^*} &= \lambda n. Push(n), \\
 var^{Com^*} &= \lambda x. Load(x), \\
 add^{Com^*} &= \lambda(c, c'). c \cdot c' \cdot Add.
 \end{aligned}$$

Let  $B$  be a set,  $\Sigma' = (S, \mathcal{I}, F)$  with

$$S = \{\text{state}\}, \quad F = \{\delta : \text{state} \rightarrow \text{state}^*\},$$

$\Sigma'(B) = (S, \mathcal{I}, F)$  with

$$\begin{aligned} S &= \{\text{state}\}, \quad F = \{ \beta : \text{state} \rightarrow B, \\ &\qquad\qquad\qquad \delta : \text{state} \rightarrow \text{state}^* \}. \end{aligned}$$

Hence the functors  $H_{\Sigma'}, H_{\Sigma'(B)} : \text{Set} \rightarrow \text{Set}$  (see  $\Sigma$ -functors) are defined as follows:

For all sets and functions  $X$ ,

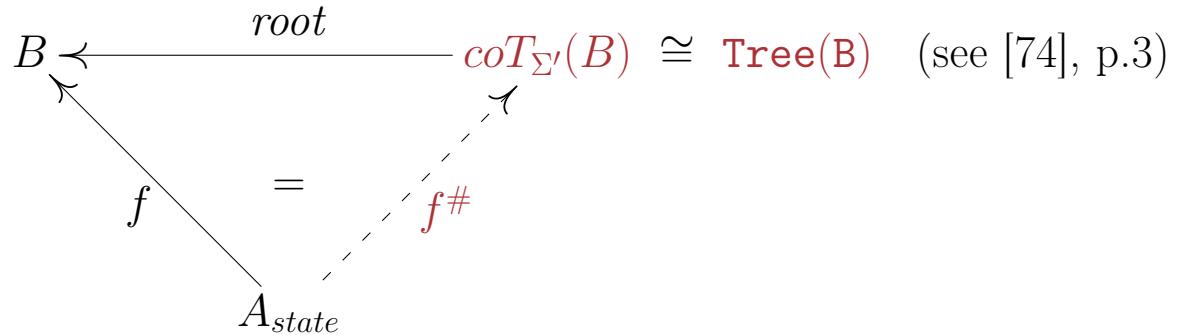
$$\begin{aligned} H_{\Sigma'}(X) &= X^*, \\ H_{\Sigma'(B)}(X) &= B \times X^*. \end{aligned}$$

Let  $A$  be a  $\Sigma'(B)$ -algebra.

Then  $\langle \beta^A, \delta^A \rangle : A \rightarrow H_{\Sigma'(B)}(A)$  is the corresponding  $H_{\Sigma'(B)}$ -coalgebra.

Let  $A$  be a  $\Sigma'$ -algebra and  $\text{co}T_{\Sigma'}(B)$  be the set of all finite and infinite trees whose nodes are labelled with  $B$ .

$\text{co}T_{\Sigma'}(B)$  is the cofree  $\Sigma'$ -algebra over  $B$ :



For all  $b \in B$ ,  $t_1, \dots, t_n \in \text{co}\mathcal{T}_{\Sigma'}(B)$ ,  $\delta^{\text{co}\mathcal{T}_{\Sigma'}(B)}(b(t_1, \dots, t_n)) = (t_1, \dots, t_n)$ .

For all  $a, a_1, \dots, a_n \in A_{state}$ ,

$$\delta^A(a) = (a_1, \dots, a_n) \Rightarrow f^\#(a) = f(a)(f^\#(a_1), \dots, f^\#(a_n)).$$

$\text{co}\mathcal{T}_{\Sigma'}(B)$  is also a **final  $\Sigma'(B)$ -algebra**:

For all  $t \in \text{co}\mathcal{T}_{\Sigma'}(B)$ ,  $\beta^{\text{co}\mathcal{T}_{\Sigma'}(B)}(t) = \text{root}(t)$ .

Let  $A$  be a  $\Sigma'(B)$ -algebra.

The unique  $\Sigma'(B)$ -homomorphism  $\text{unfold}^A : A \rightarrow \text{co}\mathcal{T}_{\Sigma'}(B)$  is defined as follows: For all  $a, a_1, \dots, a_n \in A_{state}$ ,

$$\delta^A(a) = (a_1, \dots, a_n) \Rightarrow \text{unfold}^A(a) = \beta^A(a)(\text{unfold}^A(a_1), \dots, \text{unfold}^A(a_n)).$$

$\text{unfold}^A = \text{oper}$  (see [74], p.4) for  $f = \beta^A$  and  $g = \delta^A$ .

The functor  $\text{co}\mathcal{T}_{\Sigma'} = \mathbf{Tree} : \mathbf{Set} \rightarrow \mathbf{Alg}_{\Sigma'}$  is right adjoint to the forgetful functor from  $\mathbf{Alg}_{\Sigma'}$  to  $\mathbf{Set}$ .

For all functions  $f : A \rightarrow C$ ,  $\text{co}\mathcal{T}_{\Sigma'}(f) = (f \circ \text{root})^{\#} : \text{co}\mathcal{T}_{\Sigma'}(A) \rightarrow \text{co}\mathcal{T}_{\Sigma'}(C)$ .

Let  $A$  be a  $\Sigma'(B)$ -algebra.

$$\begin{array}{ccc}
 \text{co}\mathcal{T}_{\Sigma'}(B) & \xrightarrow{\langle \beta^{\text{co}\mathcal{T}_{\Sigma'}(B)}, \delta^{\text{co}\mathcal{T}_{\Sigma'}(B)} \rangle} & H_{\Sigma'(B)}(\text{co}\mathcal{T}_{\Sigma'}(B)) \\
 \nwarrow & & \uparrow H_{\Sigma'(B)}(\text{unfold}^A) \\
 \text{unfold}^A & & \\
 \downarrow & & \\
 \vdots & & \\
 \downarrow & & \\
 A & \xrightarrow{\langle \beta^A, \delta^A \rangle} & H_{\Sigma'(B)}(A)
 \end{array}$$

The  $\Sigma'(T_\Sigma(\mathbb{Z}))$ -algebra  $T_\Sigma(\mathbb{Z})$ :

$$\begin{aligned}\beta^{T_\Sigma(\mathbb{Z})} &= id_{T_\Sigma(\mathbb{Z})} \\ \delta^{T_\Sigma(\mathbb{Z})} : T_\Sigma(\mathbb{Z}) &\rightarrow T_\Sigma(\mathbb{Z})^* =\end{aligned}$$

**trans** :: Expr Int  $\rightarrow$  [Expr Int] (see [74], p.3)

$$val(n) \mapsto \epsilon,$$

$$add(t, t') \mapsto \begin{cases} val(m + n) & \text{if } \exists m, n \in \mathbb{Z} : \\ & val(m) = t \wedge val(n) = t', \\ map(\lambda x.add(x, t'))(\delta^{T_\Sigma(\mathbb{Z})}(t)) \cdot \\ map(\lambda x.add(t, x))(\delta^{T_\Sigma(\mathbb{Z})}(t')) & \text{otherwise} \end{cases}$$

*unfold*<sup>A</sup> =  $id_{T_\Sigma(\mathbb{Z})}^\#$  = **exec** :  $T_\Sigma(\mathbb{Z}) \rightarrow coT_{\Sigma'}(T_\Sigma(\mathbb{Z}))$  (see [74], p.4)

The  $\Sigma(\mathbb{Z})$ -algebra  $A = coT_{\Sigma'}(\mathbb{Z})$ :

Let  $\Psi = (\Sigma(\mathbb{Z}), \Sigma'(\mathbb{Z}))$ . Since  $T_\Sigma(\mathbb{Z})$  is initial in  $Alg_{\Sigma'(\mathbb{Z})}$  and  $A$  is final in  $Alg_{\Sigma'(\mathbb{Z})}$ , Corollary BIIND implies that the following system of recursive  $\Psi$ -equations has a unique inductive solution in  $T_\Sigma(\mathbb{Z})$  and a unique coinductive solution in  $A$ :

Let  $n, x, y$  be variables.

$$\begin{aligned}\beta(val(n)) &= n \\ \beta(add(x, y)) &= \beta(x) + \beta(y) \\ \delta(val(n)) &= \epsilon \\ \delta(add(x, y)) &= \delta(x) \cdot \delta(y)\end{aligned}$$

Moreover, Corollary BIIND implies

$$unfold^{T_\Sigma(\mathbb{Z})} = fold^A : T_\Sigma(\mathbb{Z}) \rightarrow A.$$

**CCS** (see Calculus of Communicating Systems; [103]; [74], sections 5-8)

<b>P</b>	<b> ::= </b>	<b>N</b>	<i>-constants</i>
		$\alpha.P$	<i>-prefixing</i>
		$\sum_{i \in I} P_i$	<i>-(finite) choice</i>
		$P \mid P$	<i>-parallelism</i>
		$P \backslash \alpha$	<i>-restriction</i>
		$P[f]$	<i>-relabelling</i>

$$\begin{array}{c}
 \frac{P_j \xrightarrow{a} P'_j \quad (j \in I)}{\sum_{i \in I} P_i \xrightarrow{a} P'_j} \\
 \frac{P \xrightarrow{a} P'}{P \mid Q \xrightarrow{a} P' \mid Q} \quad \frac{Q \xrightarrow{a} Q'}{P \mid Q \xrightarrow{a} P \mid Q'} \quad \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\bar{a}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'} \\
 \\ 
 \frac{P \xrightarrow{b} P'}{P \backslash a \xrightarrow{b} P' \backslash a} \quad (a, \bar{a} \neq b) \quad \frac{P \xrightarrow{a} P'}{P[f] \xrightarrow{f(a)} P'[f]}
 \end{array}$$

FMS version ([115], p. 161)

$$\begin{array}{ll}
 a(x).P \xrightarrow{a(v)} P[v/x] & (\text{read}) \\
 \bar{a}(e).P \xrightarrow{\bar{a}(\text{val}(e))} P & (\text{write}) \\
 P_1 + P_2 \xrightarrow{a} Q \iff P_1 \xrightarrow{a} Q \vee P_2 \xrightarrow{a} Q & (\text{select}) \\
 P \mid P' \xrightarrow{a} Q \mid P' \iff P \xrightarrow{a} Q & (\text{parallelize}) \\
 P' \mid P \xrightarrow{a} P' \mid Q \iff P \xrightarrow{a} Q \\
 \\ 
 P_1 \mid P_2 \xrightarrow{\tau} Q_1 \mid Q_2 \iff P_1 \xrightarrow{a} Q_1 \wedge P_2 \xrightarrow{\bar{a}} Q_2 & (\text{communicate}) \\
 P \setminus M \xrightarrow{a} Q \setminus M \iff P \xrightarrow{a} Q \wedge a \in \text{Act} \setminus M \setminus \{\bar{b} \mid b \in M\} & (\text{restrict}) \\
 A \xrightarrow{a} Q \iff P \xrightarrow{a} Q \quad \text{if } A \text{ is defined by the equation } A = P & (\text{call})
 \end{array}$$

## A Haskell type for all recursive types

```
newtype Fix f = In (f (Fix f))
```

The functor  $P : Set \rightarrow Set$

```
data P p = Con Name
          | Pre Act p
          | Cho [p]
          | Par p p
          | Res p Act
          | Rel p (Act -> Act)
```

```
instance Functor P where
```

```
map f x = case x of
    Con n    -> Con n
    Pre a p  -> Pre a (f p)
    Cho ps   -> Cho [f p | p <- ps]
    Par p q  -> Par (f p) (f q)
    Res p a   -> Res (f p) a
    Rel p g   -> Rel (f p) g
```

**Proc** = Fix P

$\Sigma P = CCS(Act)$ , the signature for  $P$

$P$  agrees with

$$H_{\Sigma P} = \lambda A. Act + A^2 + A^2 + A \times Act + A \times Act^{Act}$$

(see  $\Sigma$ -functors).

$Proc = T_{\Sigma P}$  is an initial  $\Sigma P$ -algebra.

The functor  $T : Set \rightarrow Set$

```
data T t  = Node [(Act,t)]  
  
instance Functor T where  
  map f (Node xs) =  
    Node [(a, f t) | (a,t) <- xs]  
  
Tree = Fix T
```

$\Sigma T = \text{Proctree}(\text{Act})$ , the signature for  $T$

$T$  agrees with

$$H_{\Sigma T} = \lambda A. (\text{Act} \times A)^*$$

(see  $\Sigma$ -functors).

The  $\Sigma T$ -algebra  $\text{Tree}$  is defined as follows:

- $\text{Tree}_{\text{tree}}$  is the set of all functions  $t : \mathbb{N}^+ \rightarrow \text{Act} + 1$  such that for all  $v \in \mathbb{N}^*$ ,  $w \in \mathbb{N}^*$  and  $i > 0$ ,

$$\begin{aligned} v(i+1) \in \text{def}(t) &\Rightarrow vi \in \text{def}(t), \\ wi \in \text{def}(t) &\Rightarrow w \in \text{def}(t). \end{aligned}$$

- For all  $t \in \text{Tree}_{\text{tree}}$ ,

$$\text{denode}^{\text{Tree}}(t) = ((t(i), \lambda w.t(iw)))_{i=1}^n$$

where  $n = \max(\text{def}(t) \cap \mathbb{N})$ .

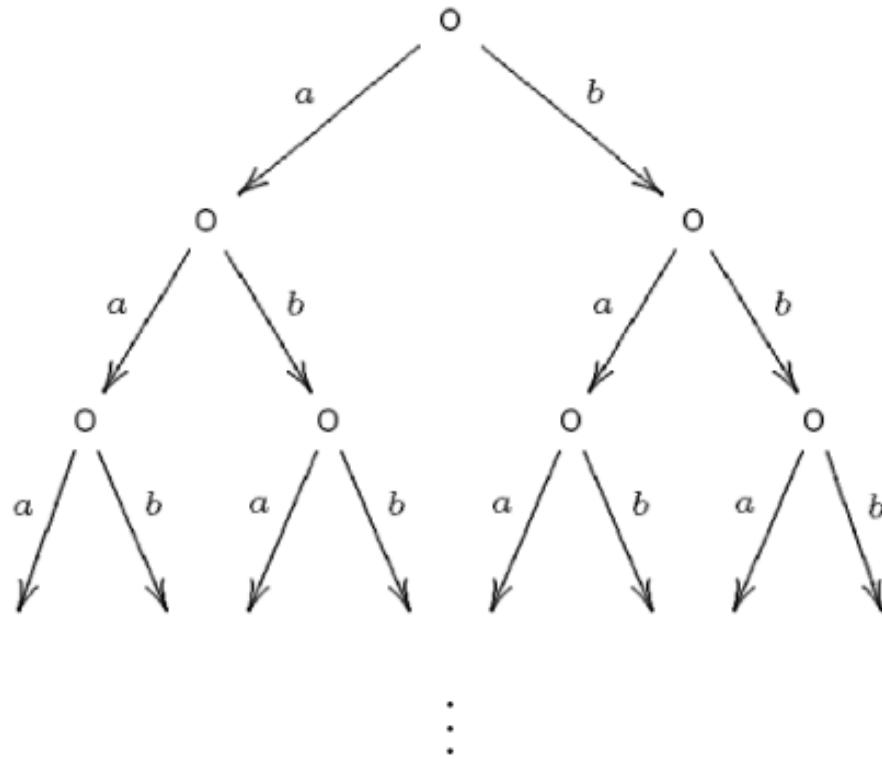


Figure 1: Transition tree for  $A = a.A + b.A$

$\text{Tree}$  is final in  $\text{Alg}_{\Sigma T}$ .

*Proof.* Let  $A$  be a  $\Sigma T$ -algebra. A function  $\text{unfold}^A : A \rightarrow \text{Tree}$  is defined as follows:

For all  $a \in A_{\text{tree}}$ ,  $i > 0$  and  $w \in \mathbb{N}^+$ ,  $\text{denode}^A(a) = ((x_i, a_i))_{i=1}^n$  implies

$$\begin{aligned}\text{unfold}^A(a)(i) &= \begin{cases} x_i & \text{if } 1 \leq i \leq n, \\ \perp & \text{otherwise,} \end{cases} \\ \text{unfold}^A(a)(iw) &= \begin{cases} \text{unfold}^A(a_i)(w) & \text{if } 1 \leq i \leq n, \\ \perp & \text{otherwise.} \end{cases}\end{aligned}$$

Let  $a \in A_{\text{tree}}$  and

$$\text{denode}^A(a) = ((x_i, a_i))_{i=1}^n. \quad (1)$$

$\text{unfold}^A$  is  $\Sigma T$ -homomorphic: By (1) and the definition of  $\text{unfold}^A$ ,

$$\max(\text{def}(\text{unfold}^A(a)) \cap \mathbb{N}) = n.$$

Hence

$$\begin{aligned}\text{denode}^{\text{Tree}}(\text{unfold}^A(a)) &\stackrel{\text{Def. }}{=} \text{denode}^{\text{Tree}}((\text{unfold}^A(a)(i), \lambda w. \text{unfold}^A(a)(iw)))_{i=1}^n, \\ &\stackrel{\text{Def. }}{=} ((x_i, \lambda w. \text{unfold}^A(a_i)(w)))_{i=1}^n = ((x_i, \text{unfold}^A(a_i)))_{i=1}^n \\ &= \text{unfold}^A(((x_i, a_i))_{i=1}^n) \stackrel{(1)}{=} \text{unfold}^A(\text{denode}^A(a)).\end{aligned}$$

*unfold<sup>A</sup>* is unique: Let  $h : A \rightarrow Tree$  be a  $\Sigma T$ -homomorphism. Then

$$denode^{Tree}(h(a)) \stackrel{h \text{ } \Sigma T-hom.}{=} h(denode^A(a)) \stackrel{(1)}{=} h(((x_i, a_i))_{i=1}^n) = ((x_i, h(a_i)))_{i=1}^n. \quad (2)$$

For all  $w \in \mathbb{N}^+$ ,

$$h(a)(w) = unfold^A(a)(w) \quad (3)$$

*Proof by induction on  $|w|$ .* For all  $1 \leq i \leq n$ ,

$$\begin{aligned} h(a)(i) &\stackrel{\text{Def. } denode^{Tree}}{=} \pi_1(\pi_i(denode^{Tree}(h(a)))) \stackrel{(2)}{=} \pi_1(\pi_i(((x_i, h(a_i)))_{i=1}^n)) \\ &= \pi_1(x_i, h(a_i)) = x_i \stackrel{\text{Def. } unfold^A}{=} unfold^A(a)(i). \end{aligned}$$

By (2) and the definition of  $denode^{Tree}$ ,

$$\max(def(h(a)) \cap \mathbb{N}) = n. \quad (4)$$

Hence for all  $i > n$  and  $w \in \mathbb{N}^*$ ,

$$h(a)(iw) \stackrel{(4)}{=} \perp \stackrel{\text{Def. } unfold^A}{=} unfold^A(a)(iw).$$

Moreover, for all  $1 \leq i \leq n$  and  $w \in \mathbb{N}^*$ ,

$$\begin{aligned} h(a)(iw) &= (\lambda w.h(a)(iw))(w) \stackrel{\text{Def. } denode^{Tree}}{=} \pi_2(\pi_i(denode^{Tree}(h(a))))(w) \\ &\stackrel{(2)}{=} \pi_2(\pi_i(((x_i, h(a_i)))_{i=1}^n))(w) = \pi_2(x_i, h(a_i))(w) = h(a_i)(w) \\ &\stackrel{\text{ind. hyp.}}{=} unfold^A(a_i)(w) \stackrel{\text{Def. } unfold^A}{=} unfold^A(a)(iw). \end{aligned}$$

Hence (3) holds true. □

Trace semantics of processes = final nondeterministic acceptor

Let  $\text{Path} = \bigcup\{def(t) \mid t \in otr(Act \times \mathbb{N}, (\Delta_{Act}, <), 1)\}$  (see chapter 3). The  $NMed^*(Act)$ -algebra  $\text{Traces}$  is defined as follows (see chapter 8):

- $\text{Traces}(\text{state}) = \mathcal{P}(\text{Path}).$
- For all  $W \subseteq \text{Path}$  and  $x \in Act$ ,

$$\delta^{\text{Traces}}(W)(x) = (\{w \in \text{Path} \mid \exists i > 0 : (x, i)w \in W\})_{i=1}^n$$

where  $n = \max\{i > 0 \mid (x, i)w \in W\}$ .

*Traces* is final in  $\text{Alg}_{N\text{Med}^*}(\text{Act})$ .

*Proof.* Let  $A$  be an  $N\text{Med}^*(\text{Act})$ -algebra. A function  $\text{unfold}^A : A \rightarrow \text{Traces}$  is defined as follows: For all  $a \in A_{\text{state}}$ ,

$$\begin{aligned}\text{unfold}^A(a) = 1 \cup \{(x, i)w \mid x \in \text{Act}, \delta^A(a)(x) = (a_1, \dots, a_n), \\ 1 \leq i \leq n, w \in \text{unfold}^A(a_i)\}.\end{aligned}$$

Let  $a \in A_{\text{state}}$ ,  $x \in \text{Act}$  and

$$\delta^A(a)(x) = (a_1, \dots, a_n). \quad (1)$$

$\text{unfold}^A$  is  $N\text{Acc}$ -homomorphic: By (1) and the definition of  $\text{unfold}^A$ ,

$$\max\{i > 0 \mid (x, i) \in \text{unfold}^A(a)\} = n.$$

Hence

$$\begin{aligned}\delta^{\text{Traces}}(\text{unfold}^A(a))(x) &\stackrel{\text{Def. }}{=} (\{w \in \text{Path} \mid (x, i)w \in \text{unfold}^A(a)\})_{i=1}^n \\ &\stackrel{\text{Def. }}{=} (\{w \in \text{Path} \mid w \in \text{unfold}^A(a_i)\})_{i=1}^n = (\text{unfold}_{\text{state}}^A(a_i))_{i=1}^n \\ &= \text{unfold}_{\text{state}^*}^A(a_1, \dots, a_n) \stackrel{(1)}{=} \text{unfold}_{\text{state}^*}^A(\delta^A(a)(x)) = \text{unfold}_{(\text{state}^*)\text{Act}}^A(\delta^A(a))(x).\end{aligned}$$

*unfold*<sup>A</sup> is unique: Let  $h : A \rightarrow Traces$  be an  $NMed^*(Act)$ -homomorphism. Then

$$\begin{aligned} \delta^{Traces}(h(a))(x) &\stackrel{h \text{- } NMed^*(Act)-\text{hom.}}{=} h_{(state^*)Act}(\delta^A(a))(x) = h_{state^*}(\delta^A(a)(x)) \\ &\stackrel{(1)}{=} h_{state^*}(a_1, \dots, a_n) = (h(a_1), \dots, h(a_n)). \end{aligned} \quad (2)$$

For all  $w \in Path$ ,

$$w \in h(a) \Leftrightarrow w \in \text{unfold}^A(a). \quad (3)$$

*Proof by induction on  $|w|$ .* By the definition of  $Traces$ , (3) holds true for  $w = \epsilon$ .

By (2) and the definition of  $\delta^{Traces}$ ,

$$\max\{i > 0 \mid (x, i) \in h(a)\} = n. \quad (4)$$

Let  $i > n$  and  $v \in Path$ . By the definition of  $\text{unfold}^A$ ,  $(x, i)v \notin \text{unfold}^A(a)$ . By (4),  $(x, i)v \notin h(a)$ . We conclude that (3) holds true for  $w = (x, i)v$ .

Moreover, for all  $1 \leq i \leq n$  and  $v \in Path$ ,

$$\begin{aligned} (x, i)v \in h(a) &\stackrel{\text{Def. } \delta^{Traces}}{\Leftrightarrow} v \in \pi_i(\delta^{Traces}(h(a))(x)) \stackrel{(2)}{\Leftrightarrow} v \in h(a_i) \\ &\stackrel{\text{ind. hyp.}}{\Leftrightarrow} v \in \text{unfold}^A(a_i) \stackrel{\text{Def. } \text{unfold}^A}{\Leftrightarrow} (x, i)v \in \text{unfold}^A(a). \end{aligned}$$

Hence (3) holds true for  $w = (x, i)v$ . □

*Proc* is a  $\Sigma T$ -algebra and *Tree* is a  $\Sigma P$ -algebra:

Let  $\Psi = (\Sigma P, \Sigma T)$ . Since *Proc* is initial in  $Alg_{\Sigma P}$  and *Tree* is final in  $Alg_{\Sigma T}$ , Corollary BIIND implies that the following system of recursive  $\Psi$ -equations has a unique inductive solution in *Proc* and a unique coinductive solution in *Tree*:

Let  $a, x, y, x', y', f$  be variables.

$$\begin{aligned}
 denode(pre(a, x)) &= (a, x) \\
 denode(cho(x, y)) &= map(denode(x)) \cdot map(denode(y)) \\
 denode(par(x, y)) &= map(\lambda(a, x').(a, par(x', y)))(denode(x)) \cdot \\
 &\quad map(\lambda(a, y').(a, par(x, y')))(denode(y)) \cdot \\
 &\quad map(\lambda((a, x'), (b, y')).(\tau, par(x', y')))) \\
 &\quad (filter(\textcolor{blue}{synch} \circ (\pi_1 \times \pi_1))(denode(x) \times denode(y))) \\
 denode(res(x, a)) &= map(\lambda(b, x').(b, res(x', a))) \\
 &\quad (filter(\lambda(b, x').\textcolor{blue}{strip}(a) \neq \textcolor{blue}{strip}(b))(denode(x))) \\
 denode(rel(x, f)) &= map(\lambda(a, x').(f(a), rel(x', f)))(denode(x))
 \end{aligned}$$

$sync : Act^2 \rightarrow 2$  and  $strip : Act \rightarrow Act$  are defined as follows: For all  $a, b \in Act$ ,

$$sync(a, b) = 1 \Leftrightarrow a = \bar{b} \vee b = \bar{a},$$

$$strip(a) = \begin{cases} b & \text{if } \exists b : a = \bar{b}, \\ a & \text{otherwise.} \end{cases}$$

The following Haskell functions `trans'` and `trans` implements the  $T$ -coalgebra

$$denode^{Proc} : Proc \rightarrow T(Proc) = (Act \times Proc)^*$$

(see [74], p.6):

```
trans'  :: Proc -> T Proc
trans' p = Node (trans p)
```

```

trans      :: Proc -> [(Act,Proc)]
trans (In x) = case x of
  Con n   -> trans (defn n)
  Pre a p -> [(a,p)]
  Cho ps   -> concat (map trans ps)
  Par p q -> [(a, par p' q) |
                (a,p') <- trans p] ++
                [(b, par p' q') |
                 (b,q') <- trans q] ++
                [(Tau, par p' q') |
                 (a,p') <- trans p,
                  (b,q') <- trans q,
                  synch a b]
  Res p a -> [(b, res p' a) |
                (b,p') <- trans p,
                strip a /= strip b]
  Rel p f -> [(f a, rel p' f) |
                (a,p') <- trans p]

```

The following Haskell function `comb` implements the  $P$ -algebra

$[pre^{Tree}, cho^{Tree}, par^{Tree}, res^{Tree}, rel^{Tree}] :$

$$Act + Tree^2 + Tree^2 + Tree \times Act + Tree \times Act^{Act} = P(Tree) \rightarrow Tree$$

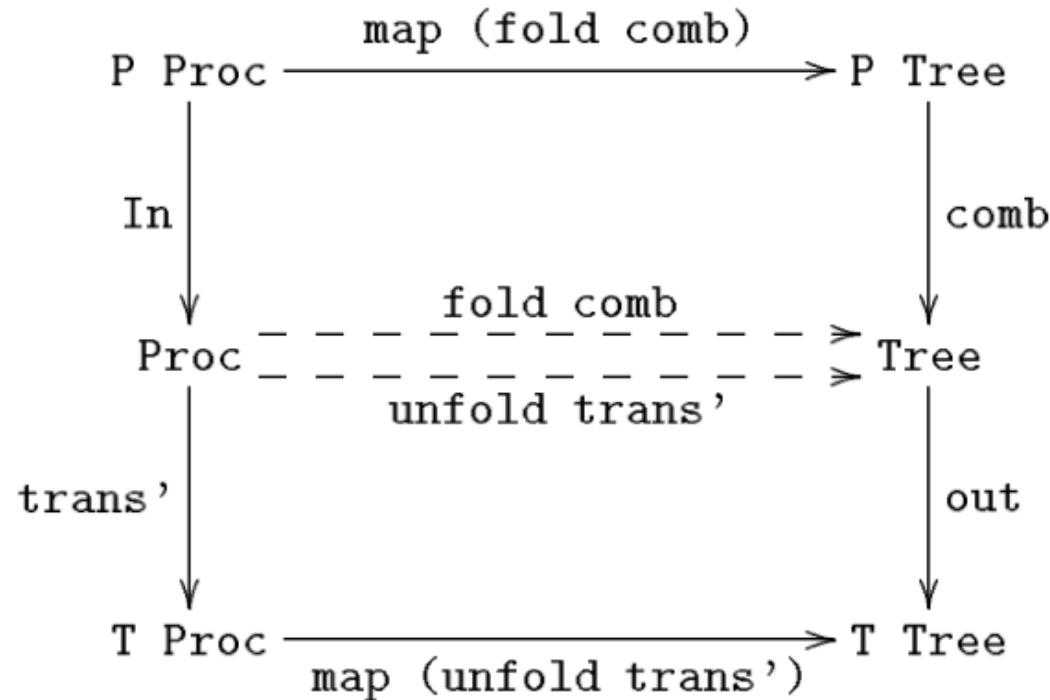
(see [74], p.7):

```
comb  :: P Tree -> Tree
comb x = In (Node (case x of
  Con n   -> denode (eval (defn n)))
  Pre a t -> [(a,t)]
  Cho ts  -> concat (map denode ts)
  Par t u -> [(a, comb (Par t' u)) |
                (a,t') <- denode t] ++
               [(b, comb (Par t u')) |
                (b,u') <- denode u] ++
               [(Tau, comb (Par t' u')) |
                (a,t') <- denode t,
                 (b,u') <- denode u,
                 synch a b]
  Res t a -> [(b, comb (Res t' a)) |
                (b,t') <- denode t,
                 strip a /= strip b]
  Rel t f -> [(f a, comb (Rel t' f)) |
                (a,t') <- denode t]))
```

```
denode :: Tree -> [(Act,Tree)]
denode (In (Node xs)) = xs
```

Moreover, Corollary BIIND implies

$$(\text{unfold trans}') = \text{unfold}^{\text{Proc}} = \text{fold}^{\text{Tree}} \text{ (= fold comb)} : \text{Proc} \rightarrow \text{Tree}.$$



In and out implement

$$[pre^{Proc}, cho^{Proc}, par^{Proc}, res^{Proc}, rel^{Proc}] :$$

$$Act + Proc^2 + Proc^2 + Proc \times Act + Proc \times Act^{Act} = P(Proc) \rightarrow Proc$$

and  $denode^{Tree} : Tree \rightarrow T(Tree) = (Act \times Tree)^*$ , respectively.

Let  $E : V \rightarrow T_{\Sigma P}(V)$  be a system of iterative  $\Sigma P$ -equations.

$E$  turns  $T_{\Sigma P}(V)$  into a  $\Sigma T$ -algebra: Let  $F$  be the set of arrows of  $\Sigma P$ .

For all  $f : e \rightarrow proc \in F$ ,  $t \in T_{\Sigma P}(V)_e$  and  $x \in V_{proc}$ ,

$$\begin{aligned} denode^{T_{\Sigma P}(V)}(ft) &= (t, f), \\ denode^{T_{\Sigma P}(V)}(x) &= denode^{T_{\Sigma P}(V)}(E(x)). \end{aligned}$$

(1)  $V \xrightarrow{inc_V} T_{\Sigma P}(V) \xrightarrow{unfold^{T_{\Sigma P}(V)}} Tree$  solves  $E$  in  $Tree$ .

*Proof.* Let  $x \in V_{proc}$ ,  $E(x) = ft$  and  $t = (t_1, \dots, t_n)$ . Hence

$$denode^{T_{\Sigma P}(V)}(x) = (t, f) \tag{2}$$

and thus for all  $i > 0$  and  $w \in \mathbb{N}_{>0}^+$ ,

$$\begin{aligned}
 & (\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V)^*(E(x))(i) = (\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V)^*(ft)(i) \\
 &= f^{\text{Tree}}((\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V)^*(t))(i) \stackrel{\text{Def. } f^{\text{Tree}}}{=} f \stackrel{\text{Def. } \text{unfold}^{T_{\Sigma P}(V), (2)}}{=} \text{unfold}^{T_{\Sigma P}(V)}(x)(\epsilon)
 \end{aligned}$$

and for all  $i \in \mathbb{N}$  and  $w \in \mathbb{N}^*$ ,

$$\begin{aligned}
 & (\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V)^*(E(x))(iw) = (\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V)^*(ft)(iw) \\
 &= f^{CT_\Sigma}((\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V)^*(u))(iw) \\
 &\stackrel{\text{Def. } f^{CT_\Sigma}}{=} \begin{cases} (\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V)^*(u_i)(w) & \text{if } u = (u_1, \dots, u_n) \text{ and } 1 \leq i \leq n \\ \perp & \text{otherwise} \end{cases} \\
 &\stackrel{\text{Def. } \text{unfold}^{T_{\Sigma P}(V), (2)}}{=} \text{unfold}^{T_{\Sigma P}(V)}(x)(iw).
 \end{aligned}$$

Therefore,  $E_{CT_\Sigma}(\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V) = (\text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V)^* \circ E = \text{unfold}^{T_{\Sigma P}(V)} \circ \text{inc}_V$ , i.e., (1) holds true. □

## Bibliography

- [1] A. Abel, B. Pientka, D. Thibodeau, A. Setzer, *Copatterns: Programming Infinite Structures by Observations*, Proc. ACM POPL (2013) 27-38
- [2] P. Aczel, *An Introduction to Inductive Definitions*, in: J. Barwise, ed., Handbook of Mathematical Logic, North-Holland (1977) 739-782
- [3] P. Aczel, J. Adamek, J. Velebil, *A Coalgebraic View of Infinite Trees and Iteration*, Proc. Coalgebraic Methods in Computer Science, Elsevier ENTCS 44 (2001) 1-26
- [4] J. Adamek, *Free algebras and automata realizations in the language of categories*, Commentat. Math Univers. Carolinae 15 (1974) 589-602
- [5] J. Adamek, *Final coalgebras are ideal completions of initial algebras*, Journal of Logic and Computation 12 (2002) 217-242
- [6] J. Adamek, *Introduction to Coalgebra*, Theory and Applications of Categories 14 (2005) 157-199
- [7] J. Adamek, *A Logic of Coequations*, Proc. CSL 2005, Springer LNCS 3634 (2005) 70-86
- [8] J. Adamek, M. Haddadi, S. Milius, *Corecursive Algebras, Corecursive Monads and Bloom Monads*, Logical Methods in Computer Science 10 (2014) 1-51

- [9] J. Adamek, D. Lücke, S. Milius, *Recursive Coalgebras of Finitary Functors*, Theor. Inform. and Appl. 41 (2007) 447–462
- [10] J. Adamek, S. Milius, L.S. Moss, *Initial algebras and terminal coalgebras: a survey*, draft of Feb. 7, 2011, TU Braunschweig
- [11] J. Adamek, H.-E. Porst, *On varieties and covarieties in a category*, Math. Structures in Computer Science 13 (2003) 201-232
- [12] J. Adamek, H.-E. Porst, *On Tree Coalgebras and Coalgebra Presentations*, Theoretical Computer Science 311 (2004) 257-283
- [13] Th. Altenkirch, *Naïve Type Theory*, in: Reflections on the Foundations of Mathematics, Springer (2019) 101-136
- [14] S. Antoy, R. Echahed, M. Hanus, *A Needed Narrowing Strategy*, Journal of the ACM 47 (2000) 776-822
- [15] M.A. Arbib, *Free dynamics and algebraic semantics*, Proc. Fundamentals of Computation Theory, Springer LNCS 56 (1977) 212-227
- [16] M.A. Arbib, E.G. Manes, *Arrows, Structures, and Functors*, Academic Press 1975
- [17] M.A. Arbib, E.G. Manes, *Parametrized Data Types Do Not Need Highly Constrained Parameters*, Information and Control 52 (1982) 139-158
- [18] E. Astesiano, H.-J. Kreowski, B. Krieg-Brückner, eds., *Algebraic Foundations of Systems Specification*, IFIP State-of-the-Art Report, Springer 1999

- [19] R. Backhouse, R. Crole, J. Gibbons, eds., *Algebraic and Coalgebraic Methods in the Mathematics of Program Construction*, Tutorial, Springer LNCS 2297 (2002)
- [20] A. Ballester-Bolinches, E. Cosme-Llópez, J. Rutten, *The dual equivalence of equations and coequations for automata*, Information and Computation 244 (2015) 49–75
- [21] M. Barr, *Terminal coalgebras in well-founded set theory*, Theoretical Computer Science 114 (1993) 299-315
- [22] M. Barr, *Terminal coalgebras for endofunctors on sets*, <ftp://ftp.math.mcgill.ca/pub/barr/pdffiles/trmclg.pdf>, McGill University, Montreal 1999
- [23] M. Barr, *Coequalizers and Free Triples*, Math. Zeitschrift 116 (1970) 307-322
- [24] M. Barr, Ch. Wells, *Category Theory*, Lecture Notes for ESSLLI, 1999
- [25] M. Barr, Ch. Wells, *Category Theory for Computing Science*, Reprints in Theory and Applications of Categories 22, 2012.
- [26] F. Bartels, *Generalised Coinduction*, Proc. CMCS 2001, Elsevier ENTCS 44 (2001)
- [27] A. Bauer, *What is algebraic about algebraic effects and handlers?*, submitted
- [28] M. Benedikt, C. Koch, *XPath Leashed*, ACM Computing Surveys 41 (2009) 3:1-3:54
- [29] R. Bird, *Introduction to Functional Programming*, Prentice Hall 1998

- [30] R. Bird, O. de Moor, *Algebra of Programming*, Prentice Hall 1997
- [31] S.L. Bloom, E.G. Wagner, Many-sorted theories and their algebras with some applications to data types, in: Maurive Nivat, John C. Reynolds: Algebraic Methods in Semantics, Cambridge University Press (1985) 133-168
- [32] L.S. Bobrow, M.A. Arbib, *Discrete Mathematics: Applied Algebra for Computer and information Science*, W.B. Saunders Company 1974
- [33] F. Bonchi, M. Bonsangue, M. Boreale, J. Rutten, A. Silva, *A coalgebraic perspective on linear weighted automata*, Information and Computation 211 (2012) 77–105
- [34] M. Bonsangue, J. Rutten, A. Silva, *An Algebra for Kripke Polynomial Coalgebras*, Proc. 24th LICS (2009) 49-58
- [35] M. Brandenburg, *Einführung in die Kategorientheorie*, Springer 2016
- [36] J.A. Brzozowski, *Derivatives of regular expressions*, Journal ACM 11 (1964) 481–494
- [37] D. Cicala, F. Honsell, M. Lenisa, *Generalized Coiteration Schemata*, Elsevier ENTCS 82 (2003)
- [38] V. Capretta, T. Uustalu, V. Vene, *Recursive coalgebras from comonads*, Information and Computation 204 (2006) 437-468

- [39] V. Capretta, T. Uustalu, V. Vene, *Corecursive algebras: A study of general structured corecursion*, Springer LNCS 5902 (2009) 84–100
- [40] R. Cockett, T. Fukushima, *About Charity*, Yellow series report 92/480/18, Dept. of Comput. Sci., Univ. of Calgary (1992)
- [41] J.R.B. Cockett, D. Spencer, *Strong categorical datatypes II: A term logic for categorical programming*, Theoretical Computer Science 139 (1995) 69-113
- [42] H. Comon et al., *Tree Automata: Techniques and Applications*, Inria 2008
- [43] C. Cîrstea, *A coalgebraic equational approach to specifying observational structures*, Theoretical Computer Science 280 (2002) 35–68
- [44] J. Cristau, C. Löding, W. Thomas, *Deterministic automata on unranked trees*, Proc. 15th FCT, Springer LNCS 3623 (2005) 68–79
- [45] A. Cunha, *Recursion Patterns as Hylomorphisms*, Technical Report DI-PURE-03.11.01, Department of Informatics, University of Minho, Portugal 2003
- [46] F. Drewes, ed., *Tree Automata*, Course notes, Umeå University, Sweden 2009
- [47] M. Droste, P. Gastin, *Weighted automata and weighted logics*, Theoretical Computer Science 380 (2007) 69–86
- [48] A. Dudenhefner, *Untersuchung und Implementierung des coinduktiven Stromkalküls*, Bachelor thesis, TU Dortmund 2011

- [49] H. Ehrig, B. Mahr, F. Cornelius, M. Große-Rhode, P. Zeitz, *Mathematisch-strukturelle Grundlagen der Informatik*, Springer 2001
- [50] M. Erwig, *Categorical Programming with Abstract Data Types*, Proc. AMAST'98, Springer LNCS 1548, 406-421
- [51] B. Fong, D.I. Spivak, *Seven Sketches in Compositionality: An Invitation to Applied Category Theory*, <https://math.mit.edu/~dspivak/teaching/sp18/7Sketches.pdf>
- [52] M.M. Fokkinga, E. Meijer, *Program Calculation Properties of Continuous Algebras*, CWI Report CS-R9104, Amsterdam 1991
- [53] J. Gibbons, G. Hutton, Th. Altenkirch, *When is a function a fold or an unfold?*, Elsevier ENTCS 44 (2001) 146-160
- [54] J.A. Goguen, R. Burstall, *Institutions: Abstract Model Theory for Specification and Programming*, J. ACM 39 (1992) 95-146
- [55] J.A. Goguen, J.W. Thatcher, E.G. Wagner, J.B. Wright, *Initial Algebra Semantics and Continuous Algebras*, J. ACM 24 (1977) 68-95
- [56] R. Goldblatt, *A Calculus of Terms for Coalgebras of Polynomial Functors*, Elsevier ENTCS 44 (2001) 161-184
- [57] G. Gottlob, C. Koch, R. Pichler, *XPath Processing in a Nutshell*, SIGMOD Record 32 (2003) 21-27

- [58] H.P. Gumm, T. Schröder, *Coalgebras of bounded type*, Math. Structures in Computer Science 12 (2002) 565-578
- [59] H.P. Gumm, *State Based Systems are Coalgebras*, Cubo - Matematica Educacional 5 (2003) 239-262
- [60] H.P. Gumm, *Equational and implicational classes of coalgebras*, Theoretical Computer Science 260 (2001) 57-69
- [61] H.P. Gumm, *Universelle Coalgebra*, in: Th. Ihringer, *Allgemeine Algebra*, Heldermann Verlag 2003
- [62] G. Gupta et al., *Infinite Computation, Co-induction and Computational Logic*, Proc. CALCO 2011, Springer LNCS 6859 (2011) 40-54
- [63] T. Hagino, *Codatatypes in ML*, J. Symbolic Computation 8 (1989) 629-650
- [64] H.H. Hansen, C. Kupke, J. Rutten, *Stream Differential Equations: Specification Formats and Solution Methods*, 2016
- [65] H.H. Hansen, J. Rutten, *Symbolic Synthesis of Mealy Machines from Arithmetic Bitstream Functions*, Scientific Annals of Computer Science (2010) 97-130
- [66] I. Hasuo, B. Jacobs, A. Sokolova, *Generic Trace Theory*, Proc. CMCS 2006, Elsevier ENTCS 164, 47-65

- [67] M. Hauhs, B. Trancón y Widemann, *Applications of Algebra and Coalgebra in Scientific Modelling Illustrated with the Logistic Map*, Elsevier ENTCS 264 (2010) 105-123
- [68] J.G. Henriksen, J. Jensen, M. Jørgensen, N. Klarlund, R. Paige, Th. Rauhe, A. Sandholm, *Mona: Monadic second-order logic in practice*, Proc. TACAS 1995, Springer LNCS 1019, 89-110
- [69] R. Hinze, *Adjoint Folds and Unfolds—An extended study*, Science of Computer Programming 78 (2013) 2108-2159
- [70] R. Hinze, *Reasoning about codata*, Third Central European Functional Programming School, Springer LNCS 6299 (2010) 42-93
- [71] R. Hinze, *Functional Pearl: Streams and Unique Fixed Points*, Proc. 13th ICFP (2008) 189-200
- [72] R. Hinze, D.W.H. James, *Proving the Unique-Fixed Point Principle Correct*, Proc. 16th ICFP (2011) 359-371
- [73] R. Hinze, N. Wu, J. Gibbons, *Conjugate Hylomorphisms*, Proc. ACM POPL (2015) 527-538
- [74] G. Hutton, *Fold and unfold for program semantics*, Proc. 3rd ICFP (1998) 280-288
- [75] G. Hutton, *A tutorial on the universality and expressiveness of fold*, J. Functional Programming 9 (1999) 355–372

- [76] B. Jacobs, *Invariants, Bisimulations and the Correctness of Coalgebraic Refinements*, Proc. Algebraic Methodology and Software Technology, Springer LNCS 1349 (1997) 276-291
- [77] B. Jacobs, *Introduction to Coalgebra*, Cambridge University Press 2017
- [78] B. Jacobs, *Exercises in Coalgebraic Specification*, Springer LNCS 2297 (2002) 237-280
- [79] B. Jacobs, *A Bialgebraic Review of Deterministic Automata, Regular Expressions and Languages*, in: K. Futatsugi et al. (eds.), Goguen Festschrift, Springer LNCS 4060 (2006) 375–404
- [80] B. Jacobs, *Trace Semantics for Coalgebras*, CMCS 2004
- [81] B. Jacobs, J. Rutten, *An introduction to (co)algebras and (co)induction*, in: D. Sangiorgi, J. Rutten (eds), Advanced topics in bisimulation and coinduction, Cambridge Univ. Press (2012) 38-99
- [82] G. Jarzembski, *A new proof of Reiterman's theorem*, Cahiers de topologie et géométrie différentielle catégoriques 35 (1994) 239-247
- [83] S. Kamin, *Final data types an their specification extension*, ACM Trans. on Prog. Lang. and Systems Comp. Syst. Sci. 5 (1983) 97-123
- [84] S.C. Kleene, *Introduction to Metamathematics*, Van Nostrand 1952

- [85] B. Klin, *Structural Operational Semantics for Weighted Transition Systems*, Mosses Festschrift, Springer LNCS 5700 (2009) 121–139
- [86] B. Klin, *Bialgebras for structural operational semantics: An introduction*, Theoretical Computer Science 412 (2011) 5043-5069
- [87] D. Kozen, *Realization of Coinductive Types*, Proc. Math. Foundations of Prog. Lang. Semantics 27, Carnegie Mellon University, Pittsburgh 2011
- [88] C. Kupke, M. Niqui, J. Rutten, *Stream Differential Equations: concrete formats for coinductive definitions*, 2011
- [89] C. Kupke, Y. Venema, *Coalgebraic automata theory: basic results*, Logical Methods in Computer Science 4 (2008) 1–43
- [90] A. Kurz, *Specifying coalgebras with modal logic*, Theoretical Computer Science 260 (2001) 119–138
- [91] A. Kurz, J. Velebil, *Relation lifting, a survey*, Journal of Logical and Algebraic Methods in Programming 85 (2016) 475–499
- [92] J. Lambek, *A fixpoint theorem for complete categories*, Math. Zeitschrift 103 (1968) 151-161
- [93] J.-L. Lassez, V.L. Nguyen, E.A. Sonenberg, *Fixed Point Theorems and Semantics: A Folk Tale*, Information Processing Letters 14 (1982) 112-116

- [94] F.W. Lawvere, *Diagonal arguments in cartesian closed categories*, Reprints in Theory and Applications of Categories 15 (2006) 1–13
- [95] D.J. Lehmann, M.B. Smyth, *Algebraic Specification of Data Types: A Synthetic Approach*, Math. Systems Theory 14 (1981) 97-139
- [96] Z. Manna, *Mathematical Theory of Computation*, McGraw-Hill 1974
- [97] E.G. Manes, M.A. Arbib *Algebraic Approaches to Program Semantics*, Springer 1986
- [98] G. Markowsky, *Chain-complete posets and directed sets with applications*, Algebra Universalis 6 (1976) 53-68
- [99] M. Marx, M. de Rijke, *Semantic Characterizations of Navigational XPath*, SIGMOD Record 34 (2005) 41-46
- [100] E. Meijer, M. Fokkinga, and R. Paterson, *Functional programming with bananas, lenses, envelopes and barbed wire*, Proc. FPCA '91, Springer LNCS 523 (1991) 124-144
- [101] E. Meijer, G. Hutton, *Bananas in Space: Extending Fold and Unfold to Exponential Types*, Proc. FPCA '95, ACM Publications (1995) 324-333
- [102] B. Milewski, *Category Theory for Programmers*, <https://bartoszmilewski.com/2014/10/28/category-theory-for-programmers-the-preface>

- [103] R. Milner, *Communication and Concurrency*, Prentice-Hall 1989
- [104] F.L. Morris, *Advice on Structuring Compilers and Proving Them Correct*, Proc. ACM POPL (1973) 144-152
- [105] T. Mossakowski, L. Schröder, M. Roggenbach, H. Reichel, *Algebraic-coalgebraic specification in CoCASL*, J. Logic and Algebraic Programming 67 (2005) 146-197
- [106] D. Orchard, *Should I use a Monad or a Comonad?*, submitted to MSFP 2012
- [107] P. Padawitz, *Church-Rosser-Eigenschaften von Graphgrammatiken und Anwendungen auf die Semantik von LISP*, Diplomarbeit, TU Berlin 1978
- [108] P. Padawitz, *Computing in Horn Clause Theories*, EATCS Monographs on Theoretical Computer Science 16, Springer-Verlag, 1988 (free copies are available from the author)
- [109] P. Padawitz, *Deduction and Declarative Programming*, Cambridge Tracts in Theoretical Computer Science 28, Cambridge University Press, 1992
- [110] P. Padawitz, *Inductive Theorem Proving for Design Specifications*, J. Symbolic Computation 21 (1996) 41-99
- [111] P. Padawitz, *Proof in Flat Specifications*, in: E. Astesiano, H.-J. Kreowski, B. Krieg-Brückner, eds., *Algebraic Foundations of Systems Specification*, IFIP State-of-the-Art Report, Springer (1999) 321-384

- [112] P. Padawitz, Swinging Types = Functions + Relations + Transition Systems, *Theoretical Computer Science* 243 (2000) 93-165
- [113] P. Padawitz, *Expander2: program verification between interaction and automation*, slides for [120], WFLP 2006
- [114] P. Padawitz, *Expander2: Two inductive proofs*, Video, TU Dortmund 2017
- [115] P. Padawitz, *Formale Methoden des Systementwurfs*, TU Dortmund 2007
- [116] P. Padawitz, *Swinging Data Types*, TU Dortmund 2009
- [117] P. Padawitz, *Dialgebraic Specification and Modeling*, TU Dortmund 2010
- [118] P. Padawitz, *Algebraic Model Checking*, in: F. Drewes, A. Habel, B. Hoffmann, D. Plump, eds., Manipulation of Graphs, Algebras and Pictures, *Electronic Communications of the EASST* Vol. 26 (2010)
- [119] P. Padawitz, *From grammars and automata to algebras and coalgebras*, Proc. CAI 2011, Springer LNCS 6742 (2011) 21-43
- [120] P. Padawitz, *Expander2 as a Prover and Rewriter*, TU Dortmund 2012
- [121] P. Padawitz, *From fixpoint to predicate co/induction and its use in standard models*, TU Dortmund 2014
- [122] P. Padawitz, *(Co)Algebraic Specification with Base Sets, Recursive and Iterative Equations*, IFIP WG 1.3 Meeting 2014

- [123] P. Padawitz, *Modeling and reasoning with I-polynomial data types*, IFIP WG 1.3 Meeting + CMS 2016
- [124] P. Padawitz, *Modellieren und Implementieren in Haskell*, TU Dortmund 2017
- [125] P. Padawitz, *Übersetzerbau (Algebraic Compiler Construction)*, TU Dortmund 2016
- [126] P. Padawitz, *Logik für Informatiker (Logic for Computer Scientists)*, TU Dortmund 2017
- [127] P. Padawitz, *From Modal Logic to (Co-)Algebraic Reasoning*, TU Dortmund 2017
- [128] D. Pattinson, *An Introduction to the Theory of Coalgebras*, Course notes for the North American Summer School in Logic, Language and Information (NASSLLI), LMU München, Germany 2003
- [129] D. Pattinson, L. Schröder, *Program Equivalence is Coinductive*, Proc. 21st LICS (2016), 337-346
- [130] D. Pavlovic and M. Escardó, *Calculus in coinductive form*, Proc. 13th LICS (1998), 408-417
- [131] S. Phillips, W.H. Wilson, G.S. Halford, *What Do Transitive Inference and Class Inclusion Have in Common? Categorical (Co)Products and Cognitive Development*, PLoS Computational Biology 5 (2009)

- [132] S. Phillips, W.H. Wilson, *Categorial Compositionality: A Category Theory Explanation for the Systematicity of Human Cognition*, PLoS Computational Biology 6 (2010)
- [133] B. Pierce, *Basic Category Theory for Computer Scientists*, MIT Press 1991
- [134] G.D. Plotkin, J. Power, *Tensors of comodels and models for operational semantics*, Elsevier ENTCS 218 (2008) 295–311
- [135] C. Pulte, *Natürliche Transformationen*, Lecture in the Proseminar *Kategorientheoretische Grundlagen*, TU Dortmund 2012
- [136] H. Reichel, *An Approach to Object Semantics based on Terminal Coalgebras*, Mathematic Structures in Computer Science 5 (1995) 129-152
- [137] H. Reichel, *Dialgebraic Logics*, Elsevier ENTCS 11 (1998) 1-9
- [138] H. Reichel, *An Algebraic Approach to Regular Sets*, in: K. Futatsugi et al., Goguen Festschrift, Springer LNCS 4060 (2006) 449-458
- [139] J. Reiterman, *The Birkhoff theorem for finite algebras*, Algebra Universalis 14 (1982) 1-10
- [140] J. Rothe, H. Tews, B. Jacobs, *The Coalgebraic Class Specification Language CCSL*, Journal of Universal Computer Science 7 (2001) 175-193
- [141] W.C. Rounds, *Mappings and Grammars on Trees*, Mathematical Systems Theory 4 (1970) 256-287

- [142] J. Rutten, *Processes as terms: non-wellfounded models for bisimulation*, Math. Struct. in Comp. Science 15 (1992) 257-275
- [143] J. Rutten, *Universal coalgebra: a theory of systems*, Theoretical Computer Science 249 (2000) 3-80
- [144] J. Rutten, *Automata and coinduction (an exercise in coalgebra)*, Proc. CONCUR '98, Springer LNCS 1466 (1998) 194–218
- [145] J. Rutten, *Automata, Power Series, and Coinduction: Taking Input Derivatives Seriously*, Proc. ICALP '99, Springer LNCS 1644 (1998) 645-654
- [146] J. Rutten, *Behavioral differential equations: a coinductive calculus of streams, automata, and power series*, Theoretical Computer Science 308 (2003) 1-53
- [147] J. Rutten, *On Streams and Coinduction*, in: *Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems*, CRM Monograph Series 23 (2004) 1-92
- [148] J. Rutten, *A coinductive calculus of streams*, Math. Struct. in Comp. Science 15 (2005) 93-147
- [149] J. Salamanca, M. Bonsangue, J. Rutten, *Equations and Coequations for Weighted Automata*, Proc. MFCS 2015, Springer LNCS 9234 (2015) 444–456
- [150] D. Sannella, A. Tarlecki, *Foundations of Algebraic Specification and Formal Software Development*, Springer 2012

- [151] D. Schwencke, *Coequational logic for accessible functors*, Information and Computation 208 (2010) 1469–1489
- [152] T. Schwentick, *XPath query containment*, SIGMOD Record 33 (2004) 101–109
- [153] K. Sen, G. Rosu, *Generating Optimal Monitors for Extended Regular Expressions*, Proc. Runtime Verification 2003, Elsevier ENTCS 89 (2003) 226-245
- [154] L. Simon, *Coinductive Logic Programming*, Ph.D. thesis, University of Texas at Dallas (2006)
- [155] A. Silva, J. Rutten, *A coinductive calculus of binary trees*, Information and Computation 208 (2010) 578–593
- [156] A. Silva, F. Bonchi, M. Bonsangue, J. Rutten, *Quantitative Kleene coalgebras*, Information and Computation 209 (2011) 822-849
- [157] J. Soto-Andrade, F.J. Varela, *Self-Reference and Fixed Points: A Discussion and an Extension of Lawvere’s Theorem*, Acta Applicandae Mathematicae 2 (1984) 1-19
- [158] D.I. Spivak, T. Giesa, E. Wood, M.J. Buehler, *Category Theoretic Analysis of Hierarchical Protein Materials and Social Networks*, [www.plosone.org](http://www.plosone.org) 2011
- [159] D.I. Spivak, R.E. Kent, *Ologs: A Categorical Framework for Knowledge Representation*, [www.plosone.org](http://www.plosone.org) 2012

- [160] D.I. Spivak, *Category Theory for the Sciences*, MIT Press 2014
- [161] A. Tarski, *A lattice-theoretical fixpoint theorem and its applications*, Pacific J. Math. 5 (1955), 285-309
- [162] W. Thomas, *Languages, Automata, and Logic*, in: *Handbook of Formal Languages*, Vol. 3: Beyond Words, Springer (1997) 389-456
- [163] W. Thomas, *Applied Automata Theory*, Course Notes, RWTH Aachen (2005)
- [164] D. Turi, G. Plotkin, *Towards a Mathematical Operational Semantics*, Proc. 12th LICS (1997) 280-291
- [165] J.W. Thatcher, E.G. Wagner, J.B. Wright, *More on Advice on Structuring Compilers and Proving Them Correct*, Theoretical Computer Science 15 (1981) 223-249
- [166] J.W. Thatcher, J.B. Wright, *Generalized Finite Automata Theory with an Application to a Decision Problem of Second-Order Logic*, Theory of Computing Systems 2 (1968) 57-81
- [167] T. Uustalu, V. Vene, *Primitive (Co)Recursion and Course-of-Value (Co)Iteration*, INFORMATICA 10 (1999) 5-26
- [168] Ph. Wadler, *Theorems for free!*, Proc. FPLCA '89, ACM Press (1989) 347-359
- [169] E.G. Wagner, J.B. Wright, J.A. Goguen, J.W. Thatcher, *Some Fundamentals of Order-Algebraic Semantics*, IBM Research Report 6020 (1976)

- [170] E.G. Wagner, J.W. Thatcher, J.B. Wright, *Free continuous theories*, IBM Research Report 6906 (1977)
- [171] E.G. Wagner, S.L. Bloom, J.W. Thatcher, *Why algebraic theories?*, in: Maurive Nivat, John C. Reynolds: Algebraic Methods in Semantics, Cambridge University Press (1985) 607-634
- [172] E.G. Wagner, *Algebraic semantics*, in: Handbook of Logic in Computer Science 3: Semantic Structures, Clarendon Press (1994) 323-393
- [173] M. Wand, *Final algebra semantics and data type extension*, J. Comp. Syst. Sci. 19 (1979) 27-44
- [174] R.F.C. Walters, *Categories and Computer Science*, Cambridge University Press 1992
- [175] J. Winter, M.M. Bonsague, J. Rutten, *Context-Free Languages, Coalgebraically*, Proc. CALCO 2011
- [176] M. Wirsing, *Structured Algebraic Specifications: A Kernel Language*, ?Theoretical Computer Science 42 (1986) 123-249
- [177] M. Wirsing, *Algebraic Specification*, in: J. van Leeuwen, ed., Handbook of Theoretical Computer Science, Elsevier (1990) 675-788
- [178] N.S. Yanofsky, *A Universal Approach to Self-Referential Paradoxes, Incompleteness and Fixed Points*, The Bulletin of Symbolic Logic 9 (2003) 362-386