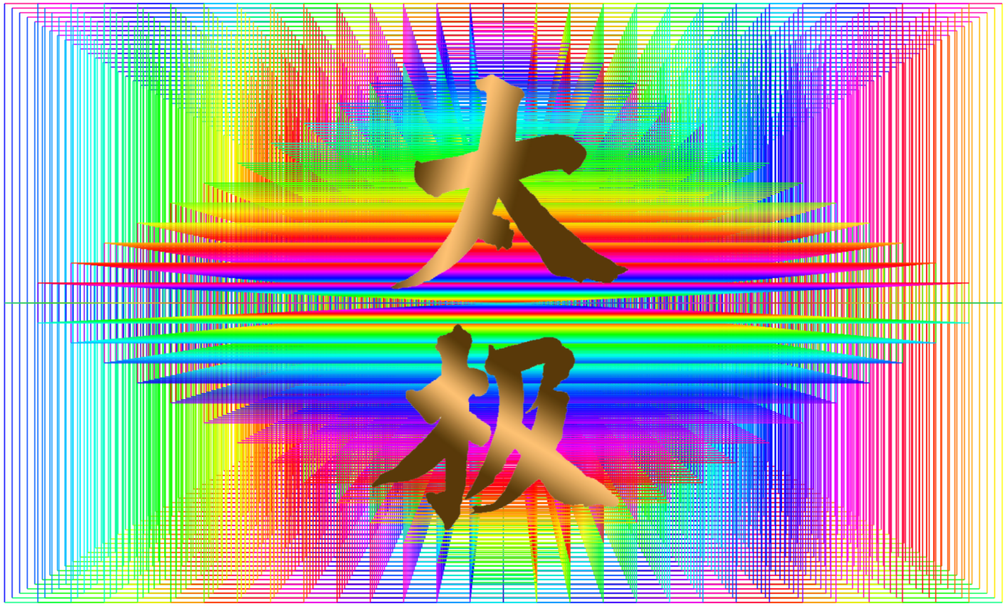


Fixpoints, Categories, and (Co)Algebraic Modeling

Peter Padawitz
TU Dortmund, Germany
August 11, 2024

(actual version: https://fldit-www.cs.tu-dortmund.de/∼peter/DialgSlides.pdf)

Contents

1 The tai chi of (co)algebraic modeling 10

2 Preliminaries 11

2.1 Products 16
2.2 Product equations 20
2.3 Product characterizations 20
2.4 Sums 23
2.5 Sum equations 27
2.6 Sum characterizations 28
2.7 Words and streams 31
2.8 Power and weighted sets 35
2.9 Labelled trees 40

3 Relations, posets and fixpoints 43

3.1 Sample CPOs 46
3.2 Fixpoints 50

4 Categories 67

4.1 From posets to categories 67

4.2 Basic definitions, examples and results . . . 71

5 Functors and natural transformations **79**

5.1 Sample functors . . . 81
5.2 The Yoneda lemma . . . 92

6 Limits and colimits **99**

6.1 Limits . . . 100
6.2 Colimits . . . 109

7 Sorted sets and types **118**

7.1 Type models . . . 122
7.2 Sorted relations . . . 126

8 Signatures **128**

8.1 Σ-arrows . . . 129
8.2 Sample constructive signatures . . . 132
8.3 Sample destructive signatures . . . 137

9 Σ-algebras **147**

9.1 Algebras and homomorphisms . . . 147
9.2 Algebras as functors and the Yoneda Lemma . . . 157
9.3 Σ-terms and -coterms . . . 161

9.4 Sample terms and coterms 164
9.5 Term and coterm algebras 171
9.6 Sample algebras 172
9.7 Product algebras 195
9.8 Sum algebras 198
9.9 Invariant algebras 201
9.10 Quotient algebras 202
9.11 Term folding 207
9.12 Term grounding 214
9.13 Sample initial algebras 215
9.14 Context-free grammars and their models 219
9.15 Removing left recursion 224
9.16 State unfolding 235
9.17 Coterm grounding 249
9.18 Sample final algebras 251
9.19 Σ-flowcharts 266
9.20 From flowcharts to terms 273

10 Σ-formulas **282**

10.1 Syntax 284
10.2 Derived terms and formulas 288

10.3 Semantics 300
10.4 Realization in Expander2 321
10.5 Automata for satisfiability 327
10.6 Institutions 336

11 Predicate specifications **338**

11.1 Syntax and semantics 338
11.2 When Kleene closures are fixpoints 361
11.3 Deduction in sequent logic 370
11.4 Rule applicability 373
11.5 Resolution and narrowing 374

12 Induction rules **379**

12.1 Fixpoint induction upon a predicate 379
12.2 Invariants and algebraic induction 383
12.3 CFGs as equations between regular expressions 389
12.4 Algebraic induction as fixpoint induction 395
12.5 Fixpoint induction upon a function 398
12.6 Invariants are monotone 399

13 Coinduction rules **403**

13.1 Fixpoint coinduction upon a predicate 403

13.2 Congruences and algebraic coinduction 410
13.3 Algebraic coinduction as fixpoint coinduction 417
13.4 Coinduction modulo constructors 422
13.5 Quotients are monotone 427
13.6 Duality of (co)resolution and (co)induction 429

14 F-algebras and -coalgebras **431**

14.1 Invariants and congruences 442
14.2 Complete categories and continuous functors 445
14.3 Initial F-algebras and final F-coalgebras 448

15 Σ-functors **461**

15.1 Functors for constructive signatures 461
15.2 Functors for destructive signatures 465
15.3 Final models of destructive non-polynomial signatures 471
15.4 From constructors to destructors 477
15.5 From destructors to constructors 481
15.6 Continuous algebras 485

16 Recursive functions **499**

16.1 Three criteria 499
16.2 Bisimulation modulo constructors 518

16.3 Sample inductive definitions . . . 521
16.4 Sample coinductive definitions . . . 544
16.5 Sample biinductive definitions . . . 570
16.6 Direct construction of a minimal acceptor of a regular language . . . 581
16.7 Guarded CFGs . . . 587
16.8 Iterative equations I . . . 595

17 Iterative equations II **596**

17.1 Algebraic theories . . . 596
17.2 Term equations . . . 597
17.3 The CPO approach for solving term equations . . . 598
17.4 The coalgebraic approach for solving term equations . . . 605
17.5 Flowchart equations . . . 620
17.6 Word acceptors . . . 627
17.7 Tree acceptors . . . 634

18 Categorical Σ-algebra **643**

18.1 Bounded functors . . . 650

19 Adjunctions **659**

19.1 Five equivalent definitions . . . 659
19.2 Identity functor . . . 669

19.3 Monoid functor . . . 669
19.4 Sequence functor . . . 672
19.5 Behavior functor . . . 674
19.6 Weighted-set functor . . . 676
19.7 Box and diamond functors . . . 678
19.8 Strongly connected components . . . 680
19.9 Reader and writer . . . 682
19.10 Cartesian closure and fixpoints . . . 683
19.11 Product and coproduct . . . 689
19.12 Term and flowchart functors . . . 693
19.13 Varieties . . . 698
19.14 Equational theories . . . 702
19.15 Coterm functors . . . 711
19.16 Covarieties . . . 713
19.17 Coequational theories . . . 717
19.18 Base algebra extensions . . . 733

20 Stream calculus 735

21 Conservative extensions 754

21.1 Constructor extensions . . . 754
21.2 Destructor extensions . . . 757

22 Abstraction and restriction **760**
22.1 Abstraction with a least congruence . . . 763
22.2 Abstraction with a greatest congruence . . . 766
22.3 Restriction with a greatest invariant . . . 772
22.4 Restriction with a least invariant . . . 777

23 λ-bialgebras **781**

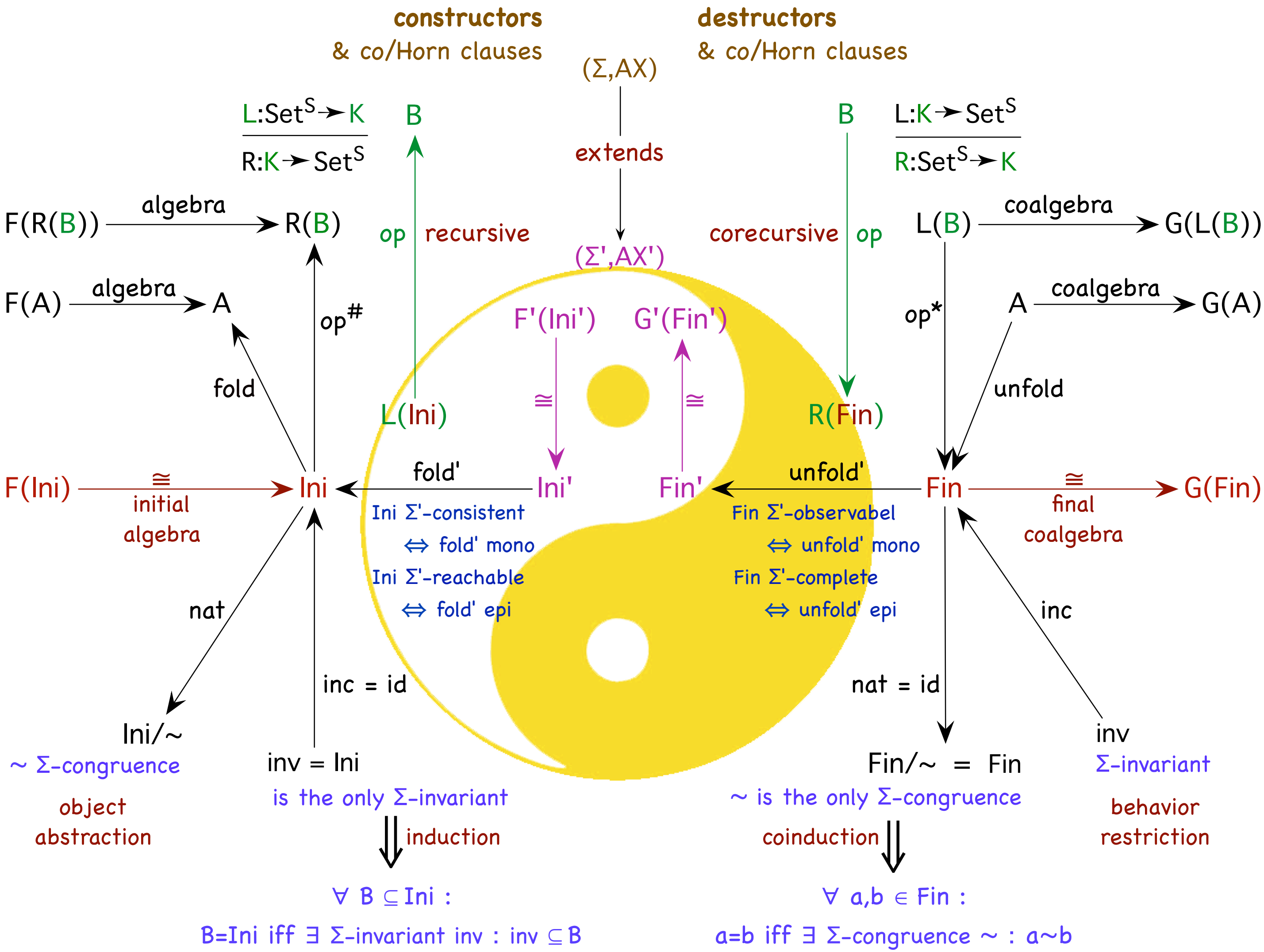
24 Monads and comonads **786**
24.1 Sample monads . . . 788
24.2 Term monads . . . 794
24.3 Sample comonads . . . 801
24.4 Coterm comonads . . . 804

25 Recursive functions as adjunctions or distributive laws **811**

26 More examples **820**
26.1 Queues . . . 820
26.2 Arithmetic expressions . . . 822
26.3 CCS . . . 831

27 Bibliography **844**

1 The tai chi of (co)algebraic modeling



2 Preliminaries

$1 = \{()\}$, $2 = \{0,1\}$, $\mathbb{N}, \mathbb{Z}, \mathbb{R}$ be the sets of natural, integer and real numbers, respectively, $\mathbb{N}_+ = \mathbb{N} \setminus \{0\}$, $[0] = \emptyset$ and for all $n > 0$, $[n] = \{1, \dots, n\}$.

Let A, B be sets.

Both $A \to B$ and B^A denote the set of total functions from A to B.

$A \multimap B$ denotes the set of partial functions from A to B.

Ω denotes the **nowhere-defined** partial function from A to B, i.e., $def(\Omega) = \emptyset$.

$|A|$ denotes the cardinality of A. $|A| < \omega$ means that A is finite. If $|A| = 1$, then A is called a **singleton** and often identified with its element.

$id_A : A \to A$ denotes the **identity** on A that maps every element of A to itself.

$\Delta_A^B = \{(a)_{b \in B} \mid a \in A\}$ denotes the B-**dimensional diagonal** of A. In particular, $\Delta_A = \Delta_A^{[2]}$.

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

A is the **domain** of f. $def(g) = \{w \in A \mid g(w) \text{ is defined}\}$ is the **domain** of g. B is the **range** of f and g. If $A = B$, then f is called an **endofunction**. $img(f) = \{f(a) \mid a \in A\}$ is called the **image** of f.

For all $b \in B$, $pre(b) = \{a \in A \mid f(a) = b\}$ is called the **pre-image** of b.

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

f is **surjective** if $img(f) = B$. f is **injective** if $ker(f) = \Delta_A$. f is **bijective** if there is a unique function $f^{-1} : B \to A$ with $f \circ f^{-1} = id_B$ and $f^{-1} \circ f = id_A$. f^{-1} is called the **inverse** of f and A and B are **isomorphic**, written as $A \cong B$.

f is bijective iff f is surjective and injective.

Let $f, g : A \to B$, I be a set (of indices), $a = (a_i)_{i \in I} \in A^I$ and $b = (b_i)_{i \in I} \in B^I$. The **function update** $f[b/a] : A \to B$ of f at a by b, the **restriction** $f|_C : C \to B$ of f to $C \subseteq A$ and the **restricted equality** $=_C \subseteq B^A \times B^A$ are defined as follows:

For all $a' \in A$, $i \in I$ and $c \in C$,

$$\begin{aligned}
f[b/a](a') &= \begin{cases} b_i & \text{if } a' = a_i, \\ f(a') & \text{otherwise,} \end{cases} \\
f|_C(c) &= f(c), \\
f =_C g &\Leftrightarrow f|_{A \setminus C} = g|_{A \setminus C}.
\end{aligned}$$

$inc_C : C \to A$ denotes the **inclusion function** that maps every element of C to itself.

Given a set A, $\mathcal{P}(A)$ denotes the **powerset** of A, i.e., the set of all subsets of A. $\mathcal{P}_+(A) =_{def} \mathcal{P}(A) \setminus \{\emptyset\}$.

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

χ is bijective. The inverse of χ is called *sat* because, given $f : A \to 2$, $sat(f)$ is the set of all $a \in A$ with $f(a) = 1$, i.e., which "satisfy" f.

For all $f : A \to A$, $f^0 =_{def} id_A$ and for all $n > 0$, $f^{n+1} =_{def} f^n \circ f$.

Let A, B, C be sets and $h : A \to B$.

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

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

$$\times_{i \in I} A_i =_{def} \{f : I \to \bigcup_{i \in I} A_i \mid \forall\, i \in I : f(i) \in A_i\}.$$

An element $f \in \bigtimes_{i\in I} A_i$ is called an ***I*-tuple** and often written as $(f(i))_{i\in I}$.

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

If for all $i, j \in I$, $i \neq j$ implies $A_i \cap A_j = \emptyset$ ("A_i and A_j are **disjoint**"), then

$$\biguplus_{i\in I} A_i \;=_{def}\; \bigcup_{i\in I} A_i.$$

Otherwise $\biguplus_{i\in I} A_i$ 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 $\iota_i(a)$.

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

The universal properties of products and sums provide a good introduction into categor(y-theoret)ical 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 6.1).

For all $f : A \to B$, $graph(f) = \{(a,b) \in A \times B \mid f(a) = b\}$ is called the **graph** of f.

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

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

$$
\begin{aligned}
p? : A &\to A + A \\
a &\mapsto if\ p(a) = 1\ then\ \iota_1(a)\ else\ \iota_2(a) \\
pair : A &\to (A \times B)^B \\
a &\mapsto \lambda b.(a, b) \\
apply : B^A \times A &\to B \\
(f, a) &\mapsto f(a) \\
curry : C^{A \times B} &\to (C^B)^A \\
f &\mapsto \lambda a.\lambda b.f(a, b) \;=\; f^B \circ pair
\end{aligned}
$$

$$
\begin{aligned}
(\times) : B^A \times D^C &\rightarrow (B \times D)^{A \times C} \\
(f, g) &\mapsto \lambda(a, c).(f(a), g(c)) \\
uncurry : (C^B)^A &\rightarrow C^{A \times B} \\
f &\mapsto \lambda(a, b).f(a)(b) = apply \circ (f \times id_B) \\
h^n : A &\rightarrow A + B \\
a &\mapsto \begin{cases} a & \text{if } n = 0 \\ h(a') & \text{if } n > 0 \wedge a' = h^{n-1}(a) \in A \\ b & \text{if } n > 0 \wedge b = h^{n-1}(a) \in B \end{cases}
\end{aligned}
$$

2.1 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, π) is called a **product of** A and written as $\prod_{i \in I} A_i$ 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 $i \in I$,

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

π_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 $f, g : B \to P$,

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

Proposition 2.1 All products of A are isomorphic to each other.

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

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

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

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

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

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

Proposition 2.2 All sets that are isomorphic to a product of A are products of A.

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

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

Projections and product extensions 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 $(f_i : B \to A_i)_{i \in I}$, $b \in B$ and $i \in I$, $\langle f_i \rangle_{i \in I}(b)(i) =_{def} f_i(b)$.

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

$$\prod_{i \in I} f_i \;=_{def}\; \langle f_i \circ \pi_i \rangle_{i \in I} : \prod_{i \in I} A_i \to \prod_{i \in I} B_i$$

is called the **product** of f.

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

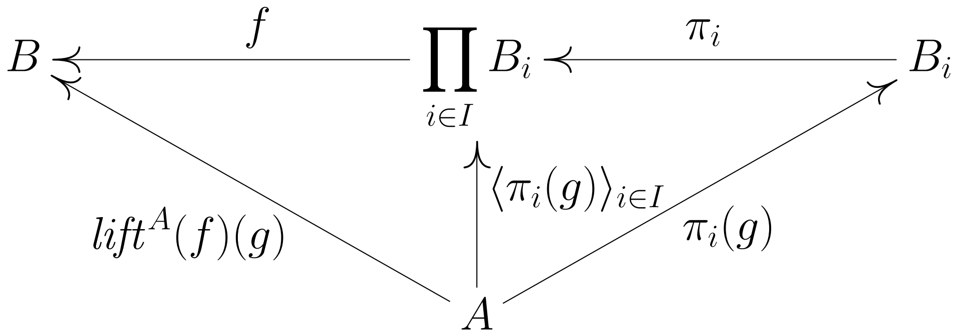
$$f^I \;=_{def}\; \textstyle\prod_{i \in I} f,$$

$$f_1 \times \cdots \times f_n \;=_{def}\; \textstyle\prod_{i \in [n]} f_i.$$

Lifting to functions from A:

$$\mathit{lift}^A : (\textstyle\prod_{i \in I} B_i \to B) \;\to\; (\textstyle\prod_{i \in I} B_i^A \to B^A)$$

$$f \;\mapsto\; \lambda g.(f \circ \langle \pi_i(g) \rangle_{i \in I}).$$



2.2 Product equations

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

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

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

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

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

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

2.3 Product characterizations

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

Proposition 2.3 (P, d) is a product of A iff for all $a, b \in P$, $a = b$ iff for all $i \in I$, $d_i(a) = d_i(b)$.

Proof. "$\Rightarrow$": Let (P, d) be a product of A, $f = \lambda x.a : 1 \to P$ and $g = \lambda x.b : 1 \to P$.

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

Proposition 2.4 (P, d) is a product of $(A_i)_{i \in I}$ iff $\langle d_i \rangle_{i \in I} : P \to \prod_{i \in I} A_i$ is iso.

Proof. The "$\Rightarrow$"-direction is shown above.

"$\Leftarrow$": Let $\langle d_i \rangle_{i \in I}$ be iso and $(f_i : B \to A_i)_{i \in I}$. 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 $f =_{def} \langle d_i \rangle_{i \in I}^{-1} \circ \langle f_i \rangle_{i \in I} : B \to P$ satisfies $d_i \circ f = f_i$.

Moreover, f is unique w.r.t. this property: Suppose that $f, g : B \to P$ 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. ❑

Proposition 2.5 (P, d) is a product of $(A_i)_{i\in I}$ iff for all $(f_i : B \to A_i)_{i\in I}$ there is a function $\langle f_i\rangle_{i\in I} : B \to P$ such that for all $i \in I$ and $f : A \to P$,

$$d_i \circ \langle f_i\rangle_{i\in I} = f_i, \quad (10)$$

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

Proof. "$\Leftarrow$": Let $(f_i : B \to A_i)_{i\in I}$ and suppose that some $\langle f_i\rangle_{i\in I} : B \to P$ satisfies (10) and (11). Let $f, g : A \to P$ 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 $f : A \to P$,

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

i.e., (11) holds true. ❑

Briefly, Proposition 2.3 tells us that equation (11) captures the uniqueness of product extensions.

Let $t, t' \in \prod_{i\in I} A_i$, $i \in I$ and $a \in A_i$. The **tuple update** $t[a/i] \in \prod_{i\in I} A_i$, the **restriction** $t|_J \subseteq \prod_{i\in J} A_i$ of t to $J \subseteq I$ and the **restricted equality** $=_J \subseteq \prod_{i\in I} A_i)^2$ are defined analogously to its function counterparts (see above): For all $k \in I$ and $j \in J$,

$$\begin{aligned} \pi_k(t[a/i]) &= \begin{cases} a & \text{if } k = i, \\ \pi_k(t) & \text{otherwise,} \end{cases} \\ \pi_j(t|_J) &= \pi_j(t'), \\ t =_J t' &\Leftrightarrow t|_{I\setminus J} = t'|_{I\setminus J}. \end{aligned}$$

2.4 Sums

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

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

$$f \circ \iota_i = f_i. \tag{12}$$

ι_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 $f, g : S \to B$,

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

Proposition 2.6 All sums of A are isomorphic to each other.

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

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

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

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

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

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

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

Proposition 2.7 All sets that are isomorphic to a sum of A are sums of A.

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

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

Injections and sum extensions 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 $(f_i : A_i \to B)_{i\in I}$, $i \in I$ and $a \in A_i$, $[f_i]_{i\in I}(a,i) =_{def} f_i(a)$.

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

$$\coprod_{i \in I} f_i \;=_{def}\; [\iota_i \circ f_i]_{i \in I} : \coprod_{i \in I} A_i \to \coprod_{i \in I} B_i$$

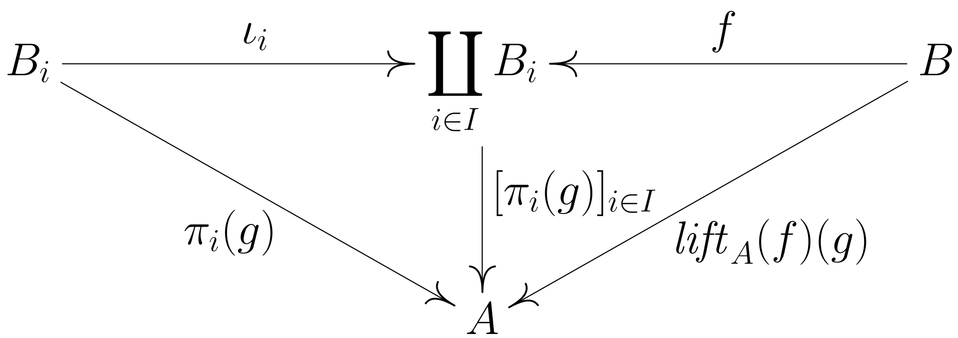
is called the **sum** or **coproduct** of f.

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

$$f \times I \;=_{def}\; \textstyle\coprod_{i \in I} f,$$
$$f_1 + \dots + f_n \;=_{def}\; \textstyle\coprod_{i \in [n]} f_i.$$

Lifting to functions to A:

$$\begin{aligned} lift_A : (B \to \textstyle\coprod_{i \in I} B_i) \;&\to\; (\textstyle\prod_{i \in I} A^{B_i} \to A^B) \\ f \;&\mapsto\; \lambda g.([\pi_i(g)]_{i \in I} \circ f). \end{aligned}$$



2.5 Sum equations

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

$$[\iota_i]_{i\in I} = id_{\coprod_{i\ inI} A_i}, \tag{16}$$

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

$$\coprod_{i\in I} g_i \circ \iota_k = \iota_k \circ g_k, \tag{18}$$

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

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

2.6 Sum characterizations

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

Proposition 2.8 (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 $f, g : S \to 2$ be defined as follows: $f = \lambda x.0$ and $g = (\lambda x.\mathit{if}\ x \in S \setminus \{a\}\ \mathit{then}\ 0\ \mathit{else}\ 1)$. 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), $f = g$. ↯ Hence $S = \bigcup_{i\in I} c_i(A_i)$.

For all $i \in I$, define $f_i : A_i \to \biguplus_{i\in I} A_i$ 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 $(f_i : A_i \to B)_{i\in I}$. Then $g : S \to B$ 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 $f : S \to B$ that satisfies (12) agrees with g. ❑

Proposition 2.9 (S, c) is a sum of $(A_i)_{i\in I}$ iff $[c_i]_{i\in I} : \coprod_{i\in I} A_i \to S$ is iso.

Proof. The "$\Rightarrow$"-direction is shown above.

"$\Leftarrow$": Let $[c_i]_{i\in I}$ be iso and $(f_i : A_i \to B)_{i\in I}$. 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 $f =_{def} [f_i]_{i\in I} \circ [c_i]_{i\in I}^{-1} : S \to B$ 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. ❏

Proposition 2.10 (S, c) is a sum of $(A_i)_{i \in I}$ iff for all $(f_i : A_i \to B)_{i \in I}$ there is $[f_i]_{i \in I} : S \to B$ such that for all $i \in I$ and $f : S \to A$,

$$[f_i]_{i \in I} \circ c_i \;=\; f_i, \tag{21}$$
$$[f \circ c_i]_{i \in I} \;=\; f. \tag{22}$$

Proof. "⇐": Let $(f_i : A_i \to B)_{i \in I}$ and suppose that some $[f_i]_{i \in I} : S \to B$ satisfies (21) and (22). Let $f, g : S \to A$ 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 $(A_i)_{i \in I}$.

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

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

i.e., (22) holds true. ❏

Briefly, Proposition 2.10 tells us that equation (22) captures the uniqueness of sum extensions.

2.7 Words and streams

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

$$
\begin{aligned}
A^+ &=_{def} \bigcup_{n>0} A^n, \\
A^* &=_{def} A^+ \cup \{\epsilon\}.
\end{aligned}
$$

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

ϵ denotes the empty word and is supposed to different from other symbols that may occur in the same context.

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

Elements of $A^{\mathbb{N}}$ are called **streams** or **infinite lists over** A.

Elements of $A^\infty =_{def} A^* \cup A^{\mathbb{N}}$ are called **colists over** A.

Functions and relations on words and streams

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

$$\begin{aligned}
head(\epsilon) &=_{def} (), \\
head(v) &=_{def} a_1, \\
head(f) &=_{def} f(0), \\
tail(\epsilon) &=_{def} (), \\
tail(v) &=_{def} \mathit{if}\ m = 1\ \mathit{then}\ \epsilon\ \mathit{else}\ (a_2, \ldots, a_m), \\
tail(f) &=_{def} \lambda n.f(n+1), \\
\epsilon \cdot \epsilon &=_{def} \epsilon, \\
w \cdot \epsilon &=_{def} w, \\
\epsilon \cdot w &=_{def} w, \\
\epsilon \cdot f &=_{def} f, \\
v \cdot w &=_{def} (a_1, \ldots, a_m, b_1, \ldots, b_n), \\
v \cdot f &=_{def} \lambda i.\mathit{if}\ i < m\ \mathit{then}\ a_{i+1}\ \mathit{else}\ f(m-i), \\
\epsilon^{-1} &=_{def} \epsilon, \\
v^{-1} &=_{def} (a_m, \ldots, a_1).
\end{aligned}$$

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

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

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

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

$L \subseteq A^*$ is **prefix closed** if all prefixes of elements of L belong to L.

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,

$$v =_{bag} w \;\Leftrightarrow_{def}\; v \text{ is a } \textbf{permutation} \text{ of } w, \text{ i.e., for all } a \in A,\ \#a(v) = \#a(w),$$
$$v =_{set} w \;\Leftrightarrow_{def}\; \text{for all } a \in A,\ \#a(v) > 0 \Leftrightarrow \#a(w) > 0.$$

Let A, B be sets and $f : A \to B$.

$$\begin{aligned} f^+ : A^+ &\to B^+ \\ (a_1, \ldots, a_n) &\mapsto (f(a_1), \ldots, f(a_n)) \end{aligned}$$

$$\begin{aligned} f^* : A^* &\to B^* \\ \epsilon &\mapsto \epsilon \\ a \cdot w &\mapsto f(a) \cdot f^*(w) \qquad a \in A,\ w \in A^* \end{aligned}$$

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

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

Proposition 2.11 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 $g : B^{A^+} \to \mathcal{C}(A, B)$ and $h : \mathcal{C}(A, B) \to B^{A^+}$ be defined as follows:

- For all $f : A^+ \to B$, $s \in A^{\mathbb{N}}$ and $n \in \mathbb{N}$, $g(f)(s)(n) = f(s(0), \ldots, s(n))$.
- For all $f' \in \mathcal{C}(A, B)$, $n \in \mathbb{N}$, $(a_1, \ldots, a_{n+1}) \in A^+$ and $s \in A^{\mathbb{N}}$,

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

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

$$h(g(f))(s(0),\ldots,s(n)) = g(f)(s)(n) = f(s(0),\ldots,s(n)),$$
$$g(h(f'))(s)(n) = h(f')(s(0),\ldots,s(n)) = f'(s)(n).$$

Hence g is bijective. ❏

2.8 Power and weighted sets

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

Let $h : A \to B$.

$\mathcal{P}(h) : \mathcal{P}(A) \to \mathcal{P}(B)$ and $\mathcal{P}_\omega(h) : \mathcal{P}_\omega(A) \to \mathcal{P}_\omega(B)$ 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 sometimes 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.

$join : \mathcal{P}(\mathcal{P}(A)) \to \mathcal{P}(A)$ maps $\mathcal{C} \subseteq \mathcal{P}(A)$ to $\bigcup \mathcal{C}$.

The **bind operator for** $\mathcal{P}$,

$$(>\!>\!=)\colon \mathcal{P}(A) \times (A \to \mathcal{P}(B)) \to \mathcal{P}(B),$$

maps (C, h) to $join(\mathcal{P}(h)(C))$.

B^A_ω denotes the set of B-weighted sets with domain A.

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

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

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

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

$M^h_\omega : M^A_\omega \to M^B_\omega$ is defined as follows: For all $f \in M^A_\omega$ and $b \in B$,

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

M_ω^h is well-defined:

For all $f \in M_\omega^A$ and $b \in B$,

$$b \in supp(M_\omega^h(f)) \;\Rightarrow \sum\{f(a) \mid a \in supp(f),\; h(a) = b\} \stackrel{(1)}{=} M_\omega^h(f)(b) \neq 0$$
$$\Rightarrow\; \exists\, a \in supp(f) : h(a) = b \;\Rightarrow\; b \in h(supp(f)).$$

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_ω^B. ❏

Given a $f \in M_\omega^A$ with $supp(f) = \{a_1, \ldots, a_n\}$ 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 $a_1, \ldots, a_n$ are regarded as variables.

This notation also allows us to define M_ω^h simply as follows: For all $m_1, \ldots, m_n \in M$,

$$M_\omega^h(\sum_{i=1}^{n} m_i \cdot a_i) = \sum_{i=1}^{n} m_i \cdot h(a_i) \tag{2}$$

(see [81], Def. 4.1.1).

$M_C^A =_{def} \{f \in M_\omega^A \mid \sum_{a\in A} f(a) \in C\}$ is called the set of C**-constrained** M**-weighted sets** with domain A and constraint φ (see [168], section 7). Hence $M_M^A = M_\omega^A$.

$M_C^h : M_C^A \to M_C^B$ denotes the restriction of M_ω^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,

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

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

$\mathcal{D}(A) =_{def} \mathbb{R}^A_{\geq 0,\{1\}}$ 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 $s = 0$, $\mathcal{D}(A)$ is a subset of $[0,1]^A_\omega$ where $[0,1]$ denotes the closed interval of real numbers between 0 and 1.

$\mathcal{D}(h) : \mathcal{D}(A) \to \mathcal{D}(B)$ is defined as $\mathbb{R}^h_{\geq 0,\{1\}}$.

M_C^A is a quotient of $(M \times A)_C^* = \{x \in (M \times A)^* \mid \sum_{i=1}^{|x|} map(\pi_1 \circ \pi_i)(x) \in C\}$.

$(M \times h)_C^* : (M \times A)_C^* \to (M \times B)_C^*$ denotes the restriction of $(M \times h)^* : (M \times A)^* \to (M \times A)^*$, which is composed of products and sums and thus defined as follows: For all $x = ((m_i, a_i))_{i=1}^n \in (M \times A)_C^*$, $(M \times h)^*(x) = ((m_i, h(a_i)))_{i=1}^n$.

$(M \times h)_C^*$ is well-defined: Let $x = ((m_i, a_i))_{i=1}^n \in (M \times A)_C^*$. Then $\sum_{i=1}^n m_i \in C$. Hence

$$(M \times h)^*(x) = ((m_i, h(a_i)))_{i=1}^n \in C.$$

2.9 Labelled trees

$ltr(A,B)$ denotes the set of **labelled trees over** (A,B), i.e., partial functions t from A^* to B such that $def(t)$ is prefix closed.

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. The label of the root of t is given by $t(\epsilon)$.

$t \in ltr(A,B)$ is often written as the prefixed set of its maximal proper subtrees:

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

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 functions $p_1,\dots,p_n : A \rightarrow 2$, $x\{a \triangleright p_1(a) \rightarrow t_1,\dots,a \triangleright p_n(a) \rightarrow t_n \mid a \in A\}$ satnds for

$$x(\bigcup_{i=1}^{n}\{a \rightarrow t_i \mid a \in A,\; p_i(a) = 1\}).$$

For all $I \subseteq A$, $b \in B$ and tree tuples $t = (t_i)_{i \in I}$, the tree $b\{i \rightarrow t_i \mid i \in I\}$ is also written as $b(t)$. In particular, $b()$ represents a leaf and is abbreviated to b.

Products, sums, words and streams yield (sets of) labelled trees:

$$\begin{aligned}
\prod_{i\in I} A_i &\cong \{()\{i \to a_i \mid i \in I\} \mid \forall\, i \in I : a_i \in A_i\},\\
\coprod_{i\in I} A_i &\cong \{i\{() \to a_i\} \mid i \in I,\ a_i \in A_i\},\\
A^+ &\cong \{n\{i \to a \mid 1 \leq i \leq n\} \mid n > 0,\ \forall\, 1 \leq i \leq n : a \in A\},\\
A^{\mathbb{N}} &\cong \{()\{n \to a \mid n \in \mathbb{N}\} \mid \forall\, n \in \mathbb{N} : a \in A\}.
\end{aligned}$$

t is **finite** if $def(t)$ is finite. t is **infinite** if $def(t)$ is infinite.

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

t is **well-founded** if for all $f \in A^{\mathbb{N}}$ there is $n \in \mathbb{N}$ such that $(f(i))_{i=1}^{n} \notin def(t)$, intuitively: t has finite depth, i.e., all paths emanating from the root are finite.

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

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

If $t \in ltr(A,B)$ is rational and finitely branching, then $def(t)$ 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 17). For instance, $t = b\{a \to t\}$ represents the tree $t = \lambda w.b$ with $def(t) = \{a\}^*$.

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

A labelled tree t over $(A\times\mathbb{N},B)$ is **ordered** if $\epsilon\in def(t)$ and for all $w\in(A\times\mathbb{N})^*$, $a\in A$ and $n\in\mathbb{N}$,

$$w(a,n+1)\in def(t)\ \Rightarrow\ w(a,n)\in def(t).$$

$otr(A\times\mathbb{N},B)$ denotes the set of all R-based labelled trees over $(A\times\mathbb{N},B)$.

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 15).

3 Relations, posets and fixpoints

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

$R^{-1} =_{def} \{(b,a) \mid (a,b) \in R\}$.

R is **reflexive** if R contains Δ_A, the 2-dimensional diagonal of A (see chapter 2).

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 $A/R =_{def} \{[a]_R \mid a \in R\}$ is called the **quotient** (set) **of** A **by** R where for all $a \in A$, $[a]_R =_{def} \{b \in A_s \mid (a,b) \in R_s\}$ is called the **equivalence class of** A **by** R.

$nat_R : A \to A/R$ denotes the **natural function** that maps every element of A to the equivalence class it belongs to.

The notion "quotient" comes from the equivalence relation $R_n \subseteq \mathbb{Z}^2$, $n > 0$, with

$$(a,b) \in R_n \quad \Leftrightarrow_{def} \quad a \ mod \ n = b \ mod \ n.$$

Obviously, $\mathbb{Z}/R_n$ is isomorphic to $\{0, \ldots, n-1\}$, i.e., to the possible remainders of divisions of integers by n.

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.

Let $C \subseteq A$. C is a **chain of** R if the restriction of R to C is a total order. $a \in A$ is a **lower bound of** C if $(a, c) \in R$ for all $c \in C$. If there is a greatest upper bound of C, it is called the **infimum of** C, denoted by $\bigsqcap C$. $a \in A$ is an **upper bound of** C if (c, a) for all $c \in C$. If there is a least upper bound of C, it is called the **supremum of** C, denoted by $\bigsqcup C$.

Chains of R^{-1} are also called **cochains of R**.

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

(A, R) is **flat** if there is $\bot_A \in A$ such that $\{\bot_A\}$ and $\{\bot_A, a\}$ with $a \in A$ are the only chains of R.

Let $(A, \leq)$ be a poset and λ 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.

(Co)chains of $\leq$ of the form $\{a_i \mid i < \lambda\}$ are called **λ-(co)chains**.

$(A, \leq)$ is **λ-complete** or a **λ-CPO** if A contains a least element $\bot_A$ w.r.t. $\leq$ and each λ-chain of $\leq$ has a supremum in A.

$(A, \leq)$ is **λ-cocomplete** or a **λ-co-CPO** if A has a greatest element $\top$ w.r.t. $\leq$ and for each λ-cochain of $\leq$ has an infimum in A.

A poset $(A, \leq)$ is **λ-bicomplete** or a **λ-bi-CPO** if A is both λ-complete and λ-cocomplete.

Note that $\geq =_{def} \leq^{-1}$ is a partial order iff $\leq$ is a partial order. However, λ-completeness w.r.t. $\geq$ need not be the same as λ-cocompleteness w.r.t. $\leq$!

3.1 Sample CPOs

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

For all sets A, B, $A \multimap B$ is a λ-CPO: The partial order is defined as follows:

For all $f, g : A \multimap B$,

$$f \leq g \iff \forall\, a \in A : f(a) = g(a) \text{ or } f(a) \text{ is undefined}$$

The least element is the nowhere-defined function Ω and every λ-chain $F \subseteq (A \multimap B)$ has a supremum:

For all $a \in A$,

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

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

The set $A \to B$ of functions from a set A to a λ-CPO $(B, \leq_B)$ is a λ-CPO. The partial order is defined argumentwise:

For all $f, g : A \to B$,

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

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

$$(\bigsqcup F)(a) \;=_{def}\; \bigsqcup_{f \in F} f(a). \qquad ❏ \tag{2}$$

Proposition 3.1 ([104], Cor. 1)

Let $(A, \leq)$ be λ-CPO. For all directed subsets B of A with $|B| \leq \lambda$, A has a supremum of B.

Proof. We show the conjecture only for $\lambda = \omega$ and refer to the proof of [104], 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 ω-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 $f : \mathbb{N} \to B$. 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 ω-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 $C = \cup_{i<\omega} B_i$: 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, $\cup_{i<\omega} B_i \subseteq B$. 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 $f : A \to B$.

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

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

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 λ-CPOs. $f : A \to B$ is **λ-continuous** if for all λ-chains C of A,

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

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

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

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

Proposition 3.2 If f is λ-continuous or λ-cocontinuous, then f is monotone. ❏

Proposition 3.3 Let f be monotone.

(1) For all λ-chains C of A, $\bigsqcup f(C) \leq f(\bigsqcup C)$.

(2) f is λ-continuous iff for all λ-chains C of A, $f(\bigsqcup C) \leq \bigsqcup f(C)$.

(3) f is λ-cocontinuous iff for all λ-cochains C of A, $\bigsqcap f(C) \leq f(\bigsqcap C)$.

(4) f is λ-continuous if A is **chain-finite**, i.e., all λ-chains of A are finite.

(5) f is λ-cocontinuous if A is **cochain-finite**, i.e., all λ-cochains of A are finite. ❏

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

3.2 Fixpoints

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 $f(a) = a$.

Theorem 3.4 (Kleene's fixpoint - or first recursion - theorem [89])

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

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

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

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

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

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

Let a be a fixpoint of f. Then a is f-closed and thus by (3), $f^n(\bot) \leq a$ for all $n \in \mathbb{N}$.

Hence $f^\infty \leq a$, i.e., f^∞ is the least fixpoint of f.

(2) Analogously. ❑

Proposition 3.5 Let $f : A \to A$ be monotone.

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

(2) If f is ω-continuous, then f^∞ is f-closed.

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

(4) If f is ω-cocontinuous, then f_∞ is f-dense.

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

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

(2) Suppose that f is ω-continuous. Then

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

(3) and (4): Analogously. ❑

Proposition 3.6 Let $f : A \to A$ be monotone. Then for all $n \in \mathbb{N}$,

$$f^\infty = f^n(\bot) \quad \Leftrightarrow \quad \forall\, i > n : f^i(\bot) = f^n(\bot), \tag{1}$$
$$f_\infty = f^n(\bot) \quad \Leftrightarrow \quad \forall\, i > n : f^i(\top) = f^n(\top). \tag{2}$$

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

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

(2) Analogously. ❏

Theorem 3.7 (fixpoint theorem for finite posets)

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

(1) If A has a least element $\bot$, then for some $n < \omega$, $f^\infty = f^n(\bot)$ is f-closed and thus by Proposition 3.5 (1), the least fixpoint of f.

(2) If A has a greatest element $\top$, then for some $n < \omega$, $f_\infty = f^n(\top)$ is f-closed and thus by Proposition 3.5 (3), the greatest fixpoint of f.

Proof. (1) Since f is monotone, induction on i implies $f^i(\bot) \leq f^{i+1}(\bot)$ for all $i \in \mathbb{N}$. Hence there is $n \in \mathbb{N}$ such that $f^n(\bot) = f^{n+1}(\bot)$ because A is finite. Therefore, $f(f^n(\bot)) = f^{n+1}(\bot) = f^n(\bot) \leq f(f^n(\bot))$ and thus $f(f^n(\bot)) = f^n(\bot)$. Induction on i implies $f^i(\bot) = f^n(\bot)$ for all $i > n$. Hence by Proposition 3.6 (1), $f^\infty = f^n(\bot)$, and thus by Proposition 3.5 (1), f^∞ is f-closed. We conclude by Theorem 3.4 (1) that f^∞ is the least fixpoint of f.

(2) Analogously. ❏

Hence, if A is finite, then the function

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

computes least and greatest fixpoints:

$$f^\infty = \textstyle\bigsqcup_{i<\omega} f^i(\bot) = \mathit{fixpt}(\leq)(f)(\bot), \qquad f_\infty = \textstyle\bigsqcap_{i<\omega} f^i(\top) = \mathit{fixpt}(\geq)(f)(\top).$$

Theorem 3.7 can be generalized from the ordinal ω to any ordinal λ:

Theorem 3.8 (Zermelo's fixpoint theorem; [2], Prop. 1.3.1; [98], Extended Folk Theorem 6; [10], Thm. 4.1.1)

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

(2) Let A be a λ-co-CPO with $|A| < \lambda$, $f : A \to A$ be monotone and $B = \{f^i(\top) \mid i < \lambda\}$ be the λ-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 $i < \lambda$, $f^i(\top) = \bigsqcap_{k \in i} f^k(\top)$.

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

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

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

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

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

We conclude from (3) that B is a λ-chain.

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

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

$$f^i(\bot) \leq b. \tag{4}$$

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

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

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

(2) Analogously. ❏

A poset A is a **complete lattice** if each subset B of A has a supremum in A.

Consequently, $\bot =_{def} \bigsqcup \emptyset = \bigsqcap A$ is the least element of $\mathcal{P}(A)$, $\top =_{def} \bigsqcup A = \bigsqcap \emptyset$ is the greatest element of $\mathcal{P}(A)$ and for all $B \subseteq A$, $\bigsqcap B =_{def} \bigsqcup \{a \in A \mid \forall\, b \in B : a \leq b\}$ is the infimum of B.

Given a set A, $\mathcal{P}(A)$ is a complete lattice with partial order $\subseteq$, supremum $\bigcup$, infimum $\bigcap$, least element $\emptyset$ and greatest element A.

Theorem 3.9 (fixpoint theorem of Knaster and Tarski [174])

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

(1) $lfp(f) =_{def} \bigsqcap \{a \in A \mid a \text{ is } f\text{-closed}\}$ is the least fixpoint of f.

(2) $f^\infty \leq lfp(f)$.

(3) If f^∞ is f-closed, then $lfp(f) \leq f^\infty$.

(4) If f^∞ is f-closed, then f^∞ is the least fixpoint of f.

(5) $gfp(f) =_{def} \bigsqcup \{a \in A \mid a \text{ is } f\text{-dense}\}$ is the greatest fixpoint of f.

(6) $gfp(f) \leq f_\infty$.

(7) If f_∞ is f-dense, then $f_\infty \le \mathit{gfp}(f)$.

(8) If f_∞ is f-dense, then f_∞ is the greatest fixpoint of f.

Proof.

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

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

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

(2) By induction on n, we obtain $f^n(\bot) \le \mathit{lfp}(f)$: $f^0(\bot) = \bot \le \mathit{lfp}(f)$ and

$$f^{n+1}(\bot) = f(f^n(\bot)) \overset{\text{ind. hyp.}}{\le} f(\mathit{lfp}(f)) \overset{(1)}{=} \mathit{lfp}(f)$$

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

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

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

(5)-(8) can be proved analogously. ❏

Compared with Theorem 3.4, Theorem 3.9 only requires monotonicity of f, but provides non-constructive fixpoints of f.

$B \subseteq A$ is **inductively defined** if B is the least fixpoint of a monotone function $F : \mathcal{P}(A) \to \mathcal{P}(A)$ and thus by Theorem 3.9 (1), the least F-closed subset of A.

$\mathbb{N}$ is inductively defined: Let $F : \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N})$ be the monotone function with

$$F(B) = \{0\} \cup \{n+1 \mid n \in B\}$$

for all $B \subseteq \mathbb{N}$. Of course, $\mathbb{N}$ is F-closed. Moreover, let $C \subseteq \mathbb{N}$ be F-closed. Assume that $C \neq \mathbb{N}$ and $n = min(\mathbb{N} \setminus C)$. Then $n \in \mathbb{N} \setminus F(C)$ because C is F-closed. Hence $n \neq 0$ and $n \neq m+1$ and thus $n-1 \neq m$ for all $m \in C$. Therefore, $n-1 \in \mathbb{N} \setminus C$, which contradicts $n = min(\mathbb{N} \setminus C)$. Consequently, $\mathbb{N} \subseteq C$ and thus $\mathbb{N}$ is the least F-closed subset of $\mathbb{N}$. ❏

The set *has*0 of streams of real numbers with at least one zero is inductively defined: Let $F : \mathcal{P}(\mathbb{R}^\mathbb{N}) \to \mathcal{P}(\mathbb{R}^\mathbb{N})$ be the monotone function with

$$F(B) = \{s \in \mathbb{R}^\mathbb{N} \mid s(0) = 0 \vee tail(s) \in B\}$$

for all $B \subseteq \mathbb{R}^{\mathbb{N}}$. Since $s(0) = 0 \vee tail(s) \in has0$ implies $s \in has0$, $has0$ is F-closed. Moreover, let $C \subseteq \mathbb{R}^{\mathbb{N}}$ be F-closed. Assume that $has0 \not\subseteq C$. Then $s \notin C$ for some $s \in has0$. Let $n = min\{k \in \mathbb{N} \mid s(k) = 0\}$. Since C is F-closed and F is monotone, $F^{n+1}(C) \subseteq C$. Hence $s \notin F^{n+1}(C)$ and thus $s(n) \neq 0$, which contradicts $n = min\{k \in \mathbb{N} \mid s(k) = 0\}$. Therefore, $has0 \subseteq C$ and we conclude that $has0$ is the least F-closed subset of $\mathbb{R}^{\mathbb{N}}$. ❏

$B \subseteq A$ is **coinductively defined** if B is the greatest fixpoint of a monotone function $F : \mathcal{P}(A) \to \mathcal{P}(A)$ and thus by Theorem 3.9 (5), the greatest F-closed subset of A.

The set $has\infty 0$ of streams of real numbers with infinitely many zeros is coinductively defined: Let $F : \mathcal{P}(\mathbb{R}^{\mathbb{N}}) \to \mathcal{P}(\mathbb{R}^{\mathbb{N}})$ be the monotone function with

$$F(B) = \{s \in has0 \mid tail(s) \in B\}$$

for all $B \subseteq \mathbb{N}$ where $has0$ is the set of streams of real numbers with at least one zero. Of course, $\mathbb{R}^{\mathbb{N}}$ is F-dense. Moreover, let $D \subseteq \mathbb{R}^{\mathbb{N}}$ be F-dense and $s \in D$. We show

$$\lambda i.s(i+n) \in has0 \text{ and } \lambda i.s(i+n+1) \in D. \quad (1)$$

by induction on n. $s \in D \subseteq F(D)$ implies $s \in has0$ and $\lambda i.s(i+1) \in D$. Hence (1) holds true for $n = 0$.

Suppose that (1) is valid. Then $\lambda i.s(i{+}n{+}1) \in D \subseteq F(D)$ and thus $\lambda i.s(i{+}n{+}1) \in has0$ and

$$\lambda i.s(i + n + 2) = \lambda i.(\lambda i.s(i + n + 1))(i + 1) \in D$$

Hence (1) holds true for $n + 1$ instead of n. But (1) implies $s \in has\infty0$. Consequently, $D \subseteq has\infty0$ and thus $has\infty0$ is the greatest F-dense subset of $\mathbb{R}^{\mathbb{N}}$. ❑

Theorem 3.10 Let $F : \mathcal{P}(A) \to \mathcal{P}(A)$ be monotone and $G : \mathcal{P}(A) \to \mathcal{P}(A)$ be defined as follows: For all $B \subseteq A$,

$$G(B) = A \setminus F(A \setminus B).$$

(1) G is monotone.

(2) $gfp(G) = A \setminus lfp(F)$.

(3) $lfp(G) = A \setminus gfp(F)$.

(4) $B \subseteq A$ is inductively defined iff $A \setminus B$ is coinductively defined.

Proof. (1) Let $B \subseteq C \subseteq A$. Hence $A \setminus C \subseteq A \setminus B$ and thus $F(A \setminus C) \subseteq F(A \setminus B)$ because F is monotone. Therefore,

$$G(B) = A \setminus F(A \setminus B) \subseteq A \setminus F(A \setminus C) = G(C).$$

(2) $$gfp(G) \overset{Thm.\ 3.9\ (5)}{=} \bigcup\{B \subseteq A \mid B \subseteq G(B)\} = \bigcup\{B \subseteq A \mid B \subseteq A \setminus F(A \setminus B)\}$$
$$= \bigcup\{B \subseteq A \mid F(A \setminus B) \subseteq A \setminus B\}$$
$$= \bigcup\{A \setminus B \mid B \subseteq A,\ F(A \setminus A \setminus B) \subseteq A \setminus A \setminus B\}$$
$$= \bigcup\{A \setminus B \mid B \subseteq A,\ F(B) \subseteq B\} = A \setminus \bigcap\{B \subseteq A \mid F(B) \subseteq B\}$$
$$\overset{Thm.\ 3.9\ (1)}{=} A \setminus lfp(F).$$

(3) Analogously.

(4) "$\Rightarrow$" follows from (2). "$\Leftarrow$" follows from (3). ❏

Let A, B be complete lattices.

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

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

Proposition 3.11 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 $f(a) \leq f(b)$. ❏

Proposition 3.12 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 $C \subseteq A$, $\bigsqcap_{a \in C} f(a) \leq f(\bigsqcap C)$. ❏

Theorem 3.13 (fixpoint induction)

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

(a) A is a complete lattice or a λ-CPO with $|A| < \lambda$, or

(b) A is an ω-CPO and f is ω-continuous.

(1) For all f-closed $a \in A$, $lfp(f) \leq a$.

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

Proof.

(1) Let (a) hold true. If A is a complete lattice, then by Theorem 3.9 (1), $lfp(f) = \bigsqcap\{a \in A \mid f(a) \leq a\} \leq a$. If A is a λ-CPO, then by transfinite induction on i, for all $i < \lambda$, $f^i(\bot) \leq a$ because f is monotone and a is f-closed. Hence by Theorem 3.8, $lfp(f) = f^{|A|}(\bot) \leq a$.

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

Hence by Theorem 3.4 (1), $\mathit{lfp}(f) = \bigsqcup_{i<\omega} f^i(\bot) \leq a$.

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

$$b =_{def} \bigsqcap_{i>0} f^i(a) \leq f^n(a) \leq a = f^0(a). \tag{3}$$

Since for all $i > 0$, $b \leq f^{i-1}(a)$ and f is monotone, $f(b) \leq f^i(a)$. 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 Theorem 3.9 (1), $\mathit{lfp}(f) = \bigsqcap\{c \in A \mid f(c) \leq c\}$. Hence (3) implies $\mathit{lfp}(f) \leq b \leq a$. If A is a λ-CPO, then by transfinite induction on i, for all $i < \lambda$, $f^{n*i}(\bot) \leq a$ because f is monotone and a is f-closed. Hence by Theorem 3.8, $\mathit{lfp}(f) = f^{|A|}(\bot) \leq a$.

Let (b) hold true. By induction on i, for all $i \in \mathbb{N}$, $f^{n*i}(\bot) \leq a$ because f is monotone and a is f-closed. Hence by Theorem 3.4 (1), $\mathit{lfp}(f) = \bigsqcup_{i<\omega} f^i(\bot) = \bigsqcup_{i<\omega} f^{n*i}(\bot) \leq a$. ❑

Theorem 3.14 (fixpoint coinduction)

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

(a) A is a complete lattice or a λ-co-CPO with $|A| < \lambda$, or

(b) A is an ω-co-CPO and f is ω-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. ❏

Theorem 3.15 (computational induction and coinduction)

(1) Let A be an ω-CPO, f^∞ be f-closed and B be an **admissible** subset of A, i.e., for all ω-chains C of A, $C \subseteq B$ implies $\bigsqcup C \in B$.

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

(2) Let A be an ω-co-CPO, $f : A \to A$ be ω-cocontinuous and B be an **co-admissible** subset of A, i.e., for all ω-cochains C of A, $C \subseteq B$ implies $\bigsqcap C \in B$.

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

Proof. (1) By assumption, for all $n \in \mathbb{N}$, $f^n(\bot) \in B$. Since f^∞ is f-closed, Theorem 3.9 (4) implies $lfp(f) = f^\infty = \bigsqcup_{n<\omega} f^n(\bot) \in B$.

(2) By assumption, for all $n \in \mathbb{N}$, $f^n(\top) \in B$. Since f_∞ is f-dense, Theorem 3.9 (8) implies $gfp(f) = f_\infty = \bigsqcap_{n<\omega} f^n(\top) \in B$. ❏

Theorem 3.16 (Noetherian induction)

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

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

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**.

4 Categories

4.1 From posets to categories

poset notion	*categorical notion*
p(artially) o(rdered) set	category
A	$\mathcal{K}$
element	object
$a \in A$	A
ordered pair	morphism
$a \leq b$	$f : A \to B$
least element	initial object
greatest element	final object
subset	diagram

$S \subseteq A$	A category $\mathcal{I}$ can be regarded as a directed graph $G = (N, E, source, target : E \to N)$ with $N = \mathcal{I}$ (nodes) and $E = Mor(\mathcal{I})$ (edges)
	A $\mathcal{K}$-diagram $\mathcal{D} : \mathcal{I} \to \mathcal{K}$ adds to G labelling functions $lab_N : N \to \mathcal{K},\ lab_E : E \to Mor(\mathcal{K}))$ with $lab_N + lab_E = \mathcal{D}$
upper bound of S	cocone of $\mathcal{D}$
lower bound of S	cone of $\mathcal{D}$
supremum (least upper bound) of S	colimit of $\mathcal{D}$
infimum (greatest lower bound) of S	limit of $\mathcal{D}$
$\mathcal{S} \subseteq \mathcal{P}(A)$	$\mathcal{S}(G)$ = class of all $\mathcal{K}$-diagrams with underlying graph G
A is $\mathcal{S}$-complete: each $S \in \mathcal{S}$ has supremum $\bigsqcup S$	$\mathcal{K}$ is $\mathcal{S}(G)$-cocomplete each $D \in \mathcal{S}(G)$ has colimit $col(D)$

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

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

A is a complete lattice:
all subsets of $\mathcal{S}$ have suprema and infima
monotone function $f : A \to B$

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

$\mathcal{K}$ is a complete and cocomplete:
all $\mathcal{K}$-diagrams have limits and colimits
functor $F : \mathcal{K} \to \mathcal{L}$

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

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

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

Knaster-Tarski Fixpoint Theorem

$f : A \to B$ is monotone $\Rightarrow$
$\bigsqcap\{a \in A \mid f(a) \leq a\}$ is least fixpoint of f
$f : A \to B$ is monotone $\Rightarrow$
$\bigsqcup\{a \in A \mid a \leq f(a)\}$ is greatest fixpoint of f

Lambek's Lemma

$\alpha : F(A) \to A$ is initial F-algebra
$\Rightarrow A$ is fixpoint of $F : \mathcal{K} \to \mathcal{K}$ (1)
$\alpha : A \to F(A)$ is final F-coalgebra
$\Rightarrow A$ is fixpoint of $F : \mathcal{K} \to \mathcal{K}$ (2)

$f : A \to B$ is $\mathcal{S}$-continuous:
$\forall\, S \in \mathcal{S} : f(\bigsqcup S) = \bigsqcup f(S)$

$F : \mathcal{K} \to \mathcal{L}$ is $\mathcal{S}(G)$-continuous:
$\forall\, D \in \mathcal{S}(G) : F(col(D)) = col(F(D))$

$f : A \to B$ is $\mathcal{S}$-cocontinuous:
$\forall\ S \in \mathcal{S} : f(\bigsqcap S) = \bigsqcap f(S)$

$F : \mathcal{K} \to \mathcal{L}$ is $\mathcal{S}(G)$-cocontinuous:
$\forall\ D \in \mathcal{S}(G) : F(lim(D)) = lim(F(D))$

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

$G = (\mathbb{N}, \{(n, n+1) \mid n \in \mathbb{N}\}, \pi_1, \pi_2)$

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

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

$F : \mathcal{K} \to \mathcal{L}$ is $\mathcal{S}(G)$-continuous,
$D = (F^n(Ini) \to F^{n+1}(Ini))_{n\in\mathbb{N}}$

$\Rightarrow$ $F(col(D)) \to col(D)$
is initial F-algebra
and thus, by (1), fixpoint of F

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

$G = (\mathbb{N}, \{(n+1, n) \mid n \in \mathbb{N}\})$

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

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

$F : \mathcal{K} \to \mathcal{L}$ is $\mathcal{S}(G)$-cocontinuous,
$D = (F^{n+1}(Fin) \to F^n(Fin))_{n\in\mathbb{N}}$

$\Rightarrow$ $lim(D) \to F(lim(D))$
is final F-coalgebra
and thus, by (2), fixpoint of F

Galois connection	adjunction
$(f : A \to B, g : B \to A)$	$F : \mathcal{K} \to \mathcal{L} \dashv G : \mathcal{L} \to \mathcal{K}$
$f(a) \le b \;\Leftrightarrow\; a \le g(b)$	$\dfrac{A \to G(B)}{F(A) \to B}$

4.2 Basic definitions, examples and results

A **(locally small) 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

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

  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 often written as $f : A \to B \in \mathcal{K}$. 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

Set and $Set_{\neq\emptyset}$ denote the categories of (nonempty) sets as objects and functions as morphisms. In terms of [172], 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.

Pfn denotes the category of sets as objects and partial functions as morphisms, i.e., for all sets A, B,

$$Pfn(A,B) = (A \multimap\!\!\rightarrow B)$$

(see, e.g., [103], section 1.3). Composition and identities are defined as follows: For all sets A, B, C,

$$\begin{array}{rcl} \circ^{Pfn} : Pfn(B,C) \times Pfn(A,B) & \rightarrow & Pfn(A,C) \\ (g,f) & \mapsto & \lambda a.\mathit{if}\ f(a)\ \mathit{is\ defined\ then}\ g(f(a))\ \mathit{else\ undefined}, \end{array}$$

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

In terms of chapter 24, $\circ^{Pfn}$ 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,

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

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

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

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

(see, e.g., [103], section 1.4).

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

$$\begin{aligned} \circ^{Mfn} : Mfn(B,C) \times Mfn(A,B) &\to Mfn(A,C) \\ (g,f) &\mapsto \lambda a.\{c \in C \mid \exists\, b \in f(a) : c \in g(b)\}. \end{aligned}$$

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

In terms of chapter 24, $\circ^{Mfn}$ is Kleisli composition and id^{Mfn} is the unit of the powerset monad.

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

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

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

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

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

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*. ❏

$f : A \to B \in \mathcal{K}$ 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.

$f : A \to B \in \mathcal{K}$ is an **embedding** and A is **embedded** in B if f is mono.

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

Lemma 4.1 Let $f : A \to B \in \mathcal{K}$ and $g : B \to C \in \mathcal{K}$.

(1) If $g \circ f$ is epi, then g is epi.

(2) If $g \circ f$ is mono, then f is mono. ❏

Lemma 4.2 Let $f : A \to C \in \mathcal{K}$, $g : A \to B \in \mathcal{K}$ and $h : B \to C \in \mathcal{K}$.

(1) If g is iso, then

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

(2) If h is iso, then

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

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

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 $ini^B : A \to B$.

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 $fin^B : B \to A$.

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.

Let $\mathcal{K}$ be a category with initial object Ini and a sum $A + B$ for all $A, B \in \mathcal{K}$.
Then for all $A \in \mathcal{K}$, $Ini + A \cong A \cong A + Ini$.

Let $\mathcal{K}$ be a category with final object Fin and a product $A \times B$ for all $A, B \in \mathcal{K}$.
Then for all $A \in \mathcal{K}$, $Fin \times A \cong A \cong A \times Fin$.

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 4.3

(1) Let A be initial in $\mathcal{K}$. All $\mathcal{K}$-monomorphisms $f : B \to A$ are isomorphisms.

(2) Let A be final in $\mathcal{K}$. All $\mathcal{K}$-epimorphisms $g : A \to B$ 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. ❑

5 Functors and natural transformations

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

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

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

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

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

Exercise 5 Show that GF is a functor. ❏

Functors preserve isomorphisms: Let $f:A\to B$ be an iso in $\mathcal{K}$ and $F:\mathcal{K}\to\mathcal{L}$ be a functor. Then

$$F(f) \circ F(f^{-1}) = F(f \circ f^{-1}) = F(id_B) = id_{F(B)},$$
$$F(f^{-1}) \circ F(f) = F(f^{-1} \circ f) = F(id_A) = id_{F(A)}.$$

$F : \mathcal{K} \to \mathcal{L}$ is **faithful** (**full**, **fully faithful**) if for all $A, B \in \mathcal{K}$, the mapping

$$\begin{aligned} F_{A,B} : \mathcal{K}(A, B) &\to \mathcal{L}(F(A), F(B)) \\ f &\mapsto F(f) \end{aligned}$$

is injective (surjective, bijective).

If F is fully faithful, then for all $A, B \in \mathcal{K}$,

$$F(A) \cong F(B) \Rightarrow A \cong B.$$

Proof. Let $F(A) \cong F(B)$. Then there is an iso $g : F(A) \to F(B)$ in $\mathcal{L}$. Since $F_{A,B}$ and $F_{B,A}$ are surjective, there are unique $f : A \to B$ and $f' : B \to A$ in $\mathcal{K}$ such that $F(f) = g$ and $F(f') = g^{-1}$. Then

$$F_{A,A}(f' \circ f) = F(f' \circ f) = F(f') \circ F(f) = g^{-1} \circ g = id_{F(A)} = F(id_A) = F_{A,A}(id_A). \quad (1)$$

Since $F_{A,A}$ is injective, (1) implies $f' \circ f = id_A$. Analogously, $f \circ f' = id_B$. Hence f is an iso in $\mathcal{K}$. ❑

5.1 Sample functors

Given an object $B \in \mathcal{L}$, the **constant functor** $const_B : \mathcal{K} \to \mathcal{L}$ 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** $Id_{\mathcal{K}} : \mathcal{K} \to \mathcal{K}$ maps each $\mathcal{K}$-object and each $\mathcal{K}$-morphism to itself.

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

Let I be a set of indices. The **diagonal functor** $\Delta^I_{\mathcal{K}} : \mathcal{K} \to \mathcal{K}^I$ 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 functors** $\times : Set^2 \to Set$ and $\prod_{i \in I} : Set^I \to Set$ map each pair (A, B) (tuple $(A_i)_{i \in I}$, respectively) of sets to its product $A \times B$ ($\prod_{i \in I} A_i$, respectively) and each pair (f, g) (tuple $(f_i : A_i \to B_i)_{i \in I}$, respectively) of functions to its product $f \times g$ ($\prod_{i \in I} f_i$, respectively; see section 2.1).

The **coproduct functors** $+ : Set^2 \to Set$ and $\coprod_{i \in I} : Set^I \to Set$ map each pair (A, B) (tuple $(A_i)_{i \in I}$, respectively) of sets to its coproduct $A + B$ ($\coprod_{i \in I} A_i$, respectively) and each pair (f, g) (tuple $(f_i : A_i \to B_i)_{i \in I}$, respectively) of functions to its coproduct $f + g$ ($\coprod_{i \in I} f_i$, respectively; see section 2.4).

Given sets I and J of indices, a functor $F : Set^I \to Set^J$ is **permutative** if for all $A \in Set^I$ 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 Set in chapter 2, 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^-_ω,
- the C**-constrained weighted-set functors** M^-_C and $(M \times \_)^*_C$,
- the **probability (distribution) functor** $\mathcal{D}$.

The **exception functor** $\_ + X : Set \to Set$ maps a set A to the set $A + X$ and a function $f : A \to B$ to the function

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

The **copower** or **writer functor** $\_ \times X : Set \to Set$ combines the identity functor with a product functor: It maps a set A to the set $A \times X$ and a function $f : A \to B$ to the function

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

Let X represent a set of states.

The **state functor** (also called **store** or **side-effects functor**; see [112])

$$(\_ \times X)^X : Set \to Set$$

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

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

The **costate functor**

$$(\_^X) \times X : Set \to Set$$

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

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

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** $LT(X) : Set \to Set$ maps a set A to the set of labelled trees over (X, A) and a function $f : A \to B$ to the function

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

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

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

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

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

Let $X \in \mathcal{K}$. Reader and coreader functors can be generalized to the **partial hom-functors** $\mathcal{K}(X, \_) : \mathcal{K} \to Set$ and $\mathcal{K}(\_, X) : \mathcal{K}^{op} \to Set$ that are defined on morphisms as follows: For all $f : A \to B \in \mathcal{K}$,

$$\mathcal{K}(X, f) =_{def} \lambda h : X \to A.f \circ h : \mathcal{K}(X, A) \to \mathcal{K}(X, B),$$
$$\mathcal{K}(f, X) =_{def} \lambda h : B \to X.h \circ f : \mathcal{K}(B, X) \to \mathcal{K}(A, X).$$

$$\begin{array}{ccc} A & & \mathcal{K}(X,A) \\ f \downarrow & \overset{\mathcal{K}(X,\_)}{\mapsto} & \downarrow \mathcal{K}(X,f) \\ A & & \mathcal{K}(X,B) \end{array} \qquad\qquad \begin{array}{ccc} B & & \mathcal{K}(B,X) \\ f \uparrow & \overset{\mathcal{K}(\_,X)}{\mapsto} & \downarrow \mathcal{K}(f,X) \\ A & & \mathcal{K}(A,X) \end{array}$$

The functor $\mathcal{K}(\_, X)$ is a presheaf because it maps to Set.

For applying the $\mathcal{K}(X, \_)$ and $\mathcal{K}(\_, X)$ in parallel, their domain categories are combined to the product category $\mathcal{K}^{op} \times \mathcal{K}$.

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

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

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

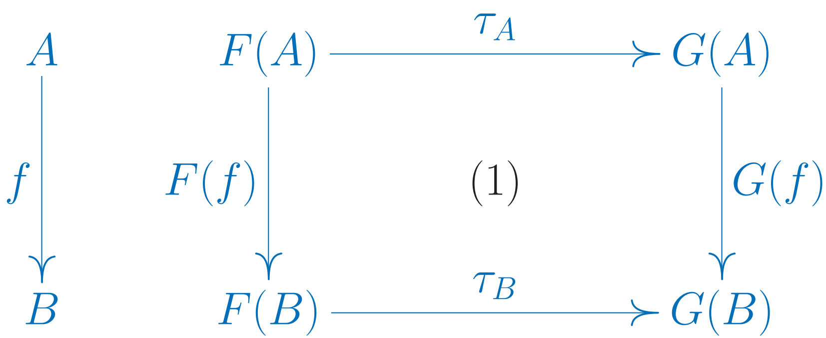
$$\begin{array}{ccc} (A, C) & & \mathcal{K}(A, C) \\ f \uparrow \; \downarrow g & \overset{\mathcal{K}(\_,\_)}{\mapsto} & \downarrow \mathcal{K}(f, g) \\ (B, D) & & \mathcal{K}(B, D) \end{array}$$

Cat denotes the category with categories as objects and functors as morphisms.

Given two functors $F, G : \mathcal{K} \to \mathcal{L}$, a **natural transformation**

$$\tau = (\tau_A : F(A) \to G(A))_{A \in \mathcal{K}} : F \to G$$

is a tuple $\mathcal{L}$-morphisms such that for all $\mathcal{K}$-morphisms $f : A \to B$ diagram (1) commutes:



If for all $A \in \mathcal{K}$, τ_A is an isomorphism, then $\tau : F \to G$ is a **natural equivalence** and F and G are **naturally equivalent** or **iso**(**morphic**), written as $F \cong G$.

Examples

1. The Haskell function `concat::[[a]]->[a]` is a natural transformation from the composition of the list functor with the list functor to the list functor, i.e., for all sets A, $F(A) = (A^*)^*$ and $G(A) = A^*$.

$concat_A : F(A) \to G(A)$ is inductively defined as follows: For all $v \in A^*$ and $w \in (A^*)^*$,

$$\begin{aligned} concat_A(\epsilon) &= \epsilon, \\ concat_A(vw) &= f(v) \cdot concat_A(w). \end{aligned}$$

concat is also the multiplication of the list monad (see chapter 24).

Exercise 6 Show that $\tau = concat$ satisfies (1).

2. The Haskell function `uncurry(++)::([a],[a])->[a]` is a natural transformation from the functor composition

$$Set \xrightarrow{\_^*} Set \xrightarrow{\Delta^2_{Set}} Set^2 \xrightarrow{\times} Set$$

to the list functor, i.e., for all sets A, $F(A) = (\times)(\Delta^2_{Set}(A^*))$ and $G(A) = A^*$.

$uncurry(++)_A : F(A) \to G(A)$ is defined as follows: For all $v, w \in A^*$,

$$uncurry(++)_A(v, w) \;=\; v \cdot w.$$

Exercise 7 Show that $\tau = uncurry(++)$ satisfies (1).

3. Exercise 8 $\tau : LT(X) \to \mathcal{P}$ defined by $\tau_A(t) = \{t(w) \mid w \in def(t)\}$ for all sets A and $t \in ltr(X, A)$ satisfies (1).

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

$$st^{T,A} : T \circ \_^A \;\to\; \_^A \circ T$$

of T and A is defined as follows (see [83], p. 380):

For all sets B, $g \in T(B^A)$ and $a \in A$,

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

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

$st^{T,A}$ is a natural transformation, i.e., for all $h : B \to C$, diagram (2) commutes:

$$\begin{array}{ccccccc} B^A & & T(B^A) & \xrightarrow{st_B^{T,A}} & T(B)^A & & B \\ \downarrow h^A & & \downarrow T(h^A) & (2) & \downarrow T(h)^A & & \downarrow h \\ C^A & & T(C^A) & \xrightarrow{st_C^{T,A}} & T(C)^A & & C \end{array}$$

Proof. At first, we show:

$$(\lambda f.f(a)) \circ h^A = \lambda f.h(f(a)). \tag{3}$$

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

$$(T(h)^A \circ st_B^{T,A}(g))(a) = (T(h)^A \circ \lambda a.T(\lambda f.f(a))(g))(a) = T(h)^A(T(\lambda f.f(a))(g))$$
$$= (T(h) \circ T(\lambda f.f(a)))(g) = T(h \circ \lambda f.f(a))(g) = T(\lambda f.h(f(a)))(g),$$
$$st_C^{T,A}(T(h^A)(g))(a) = T(\lambda f.f(a))(T(h^A)(g)) = (T(\lambda f.f(a)) \circ T(h^A))(g)$$
$$= T((\lambda f.f(a)) \circ h^A)(g) \overset{(3)}{=} T(\lambda f.h(f(a)))(g)$$

❏

Given two categories $\mathcal{K}$ and $\mathcal{L}$, $Fun(\mathcal{K}, \mathcal{L})$ denotes the category of functors from $\mathcal{K}$ to $\mathcal{L}$ as objects and all natural transformations between such functors as objects.

By [173], Theorem 11.1, for every small category $\mathcal{K}$, the set $Fun(\mathcal{K}^{op}, Set)$ of presheaves (see section 5.1) is *Cartesian closed* (see section 19.10).

5.2 The Yoneda lemma

The functors

$$\begin{aligned} H^* : \mathcal{K}^{op} &\to Fun(\mathcal{K}, Set) \\ X \in \mathcal{K} &\mapsto \mathcal{K}(X, \_) \\ f \in \mathcal{K}(B, A) &\mapsto (\lambda g.g \circ f : \mathcal{K}(A, X) \to \mathcal{K}(B, X))_{X \in \mathcal{K}} \end{aligned}$$

and

$$\begin{aligned} H_* : \mathcal{K} &\to Fun(\mathcal{K}^{op}, Set) \\ X \in \mathcal{K} &\mapsto \mathcal{K}(\_, X) \\ f \in \mathcal{K}(A, B) &\mapsto (\lambda g.f \circ g : \mathcal{K}(X, A) \to \mathcal{K}(X, B))_{X \in \mathcal{K}} \end{aligned}$$

are called **Yoneda embeddings**.

Indeed, H^* and H_* are injective: Suppose that A, B are different objects of $\mathcal{K}$. Then $\mathcal{K}(A, A)$ and $\mathcal{K}(B, A)$ as well as $\mathcal{K}(A, A)$ and $\mathcal{K}(A, B)$ are disjoint because morphisms have unique sources and targets. Hence both $H^*(A) \neq H^*(B)$ and $H_*(A) \neq H_*(B)$.

Lemma 5.1

Let $\mathcal{K}$ be a small category. For all functors $F : \mathcal{K} \to Set$, $G : \mathcal{K}^{op} \to Set$ and $A \in \mathcal{K}$,

$$Fun(\mathcal{K}, Set)(\mathcal{K}(A, \_), F) \cong F(A), \tag{5}$$
$$Fun(\mathcal{K}^{op}, Set)(\mathcal{K}(\_, A), G) \cong G(A). \tag{6}$$

Proof. (5) The functions

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

and

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

are defined as follows:

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

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

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

for all $B \in \mathcal{K}$ and $h : A \to B \in \mathcal{K}$.

Φ is iso with inverse Ψ: 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 $\tau : \mathcal{K}(A, \_) \to F$, $B \in \mathcal{K}$ and $h : A \to B \in \mathcal{K}$,

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

(6) Analogously. ❏

Corollary 5.2 Let $\mathcal{K}$ be a small category. For all objects $A, B \in \mathcal{K}$,

$$A \cong B \quad \Leftrightarrow \quad \mathcal{K}(A, \_) \cong \mathcal{K}(B, \_), \tag{7}$$

$$A \cong B \quad \Leftrightarrow \quad \mathcal{K}(\_, A) \cong \mathcal{K}(\_, B). \tag{8}$$

Proof. (7): "$\Rightarrow$": Let $A \cong B$ and $C \in \mathcal{K}$. Since $\mathcal{K}(\_, C)$ is a functor, $\mathcal{K}(A, C) \cong \mathcal{K}(B, C)$. Hence $\mathcal{K}(A, \_)$ and $\mathcal{K}(B, \_)$ are naturally equivalent.

"$\Leftarrow$" ([35], 5.2.8): At first, we show—following the proof given in https://de.wikipedia.org/wiki/Lemma_von_Yoneda—that the Yoneda embedding H^* is fully faithful, i.e., for all $A, B \in \mathcal{K}$,

$$H^*_{A,B} : \mathcal{K}(B,A) = \mathcal{K}^{op}(A,B) \;\to\; \begin{cases} Fun(\mathcal{K}, Set)(H^*(A), H^*(B)) \\ = Fun(\mathcal{K}, Set)(\mathcal{K}(A,\_), \mathcal{K}(B,\_)) \end{cases}$$
$$f \;\mapsto\; H^*(f)$$

is bijective. By the proof of Lemma 5.1,

$$\begin{aligned} \Phi : Fun(\mathcal{K}, Set)(\mathcal{K}(A,\_), \mathcal{K}(B,\_)) &\to \mathcal{K}(B,A) \\ \tau : \mathcal{K}(A,\_) \to \mathcal{K}(B,\_) &\mapsto \tau_A(id_A) \end{aligned} \tag{9}$$

is iso. Since for all $f : B \to A \in \mathcal{K}$,

$$\Phi(H^*_{A,B}(f)) = \Phi(H^*(f)) \stackrel{(9)}{=} H^*(f)_A(id_A) = (\lambda g.g \circ f)(id_A) = id_A \circ f = f \tag{10}$$

and Φ is a retraction, $H^*_{A,B} = \Phi^{-1}$. Hence for all natural transformations $\tau : \mathcal{K}(A,\_) \to \mathcal{K}(B,\_)$,

$$H^*_{A,B}(\Phi(\tau)) = \Phi^{-1}(\Phi(\tau)) = \tau. \tag{11}$$

By (10) and (11), $H^*_{A,B}$ is bijective and thus by (1),

$$H^*(A) = \mathcal{K}(A,\_) \cong \mathcal{K}(A,\_) = H^*(B) \text{ implies } A \cong B.$$

(8) Analogously with H_* instead of H^*. ❑

Mazur ([106], section 14 ff.) and Brandenburg ([35], 5.2.11.3) interpret (7) and (8) in a very foundational, even sociological way: Each "individual" (object) $A \in \mathcal{K}$ is determined (up to isomorphism) by its "contacts to the environment", given by $\mathcal{K}(A, \_)$ and $\mathcal{K}(\_, A)$.

Functors $F : \mathcal{K} \to Set$ 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., [172], sections 4.5 and 7.2.1). In fact, "database schema" and "schema instance" correspond to "signature" and "algebra", respectively (see chapters 12, 13 and 14).

Corollary 5.2 allows to prove isomorphisms in all categories with certain constraints by arguing in the category Set, which mostly provides more structure to be used the proof. For instance, the tedious direct proof that all Cartesian closed categories with coproducts satisfy the distributive law

$$A \times (B + C) \;\cong\; (A \times B) + (A \times C)$$

can be simplified considerably by showing the equivalent bijection

$$\mathcal{K}(A \times (B + C), X) \;\cong\; \mathcal{K}((A \times B) + (A \times C), X)$$

for all $X \in \mathcal{K}$ (see [19], Proposition 8.6, or [172], Exercise 7.2.1.22).

$A \in \mathcal{K}$ **represents** a covariant or contravariant functor $F : \mathcal{K} \to \mathit{Set}$ if F is naturally equivalent to $\mathcal{K}(A, \_)$ or $\mathcal{K}(\_, A)$, respectively.

Examples

1 represents $\mathit{Id}_{\mathit{Set}}$ (see [146], section 4.2.1).

The monoid $(\mathbb{N}, \{0, (+)\})$ represents the forgetful functor $U : \mathit{Monoid} \to \mathit{Set}$ (see 19.3 and [146], section 4.2.2).

Further examples are given in sections 19.12 and 19.15: Term sets and coterm sets are representable.

Compositions of natural transformations with functors

- Let $F, G : \mathcal{K} \to \mathcal{L}$, $\tau : F \to G$ and $H : \mathcal{L} \to \mathcal{M}$.
  Then $H\tau : HF \to HG$ and for all $A \in \mathcal{K}$, $(H\tau)_A = H(\tau_A) : HF(A) \to HG(A)$.

- Let $F : \mathcal{K} \to \mathcal{L}$, $G, H : \mathcal{L} \to \mathcal{M}$ and $\tau : G \to H$.
  Then $\tau F : GF \to HF$ and for all $A \in \mathcal{K}$, $(\tau F)_A = \tau_{F(A)} : GF(A) \to HF(A)$.

- *Vertical Composition*
  Let $F, G, H : \mathcal{K} \to \mathcal{L}$, $\tau : F \to G$ and $\eta : G \to H$.
  Then $\eta\tau : F \to H$ and for all $A \in \mathcal{K}$, $(\eta\tau)_A = \eta_A \circ \tau_A : F(A) \to H(A)$.

- *Horizontal Composition*
  Let $F, G : \mathcal{K} \to \mathcal{L}$, $\tau : F \to G$, $F', G' : \mathcal{L} \to \mathcal{M}$ and $\tau' : F' \to G'$. Then

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

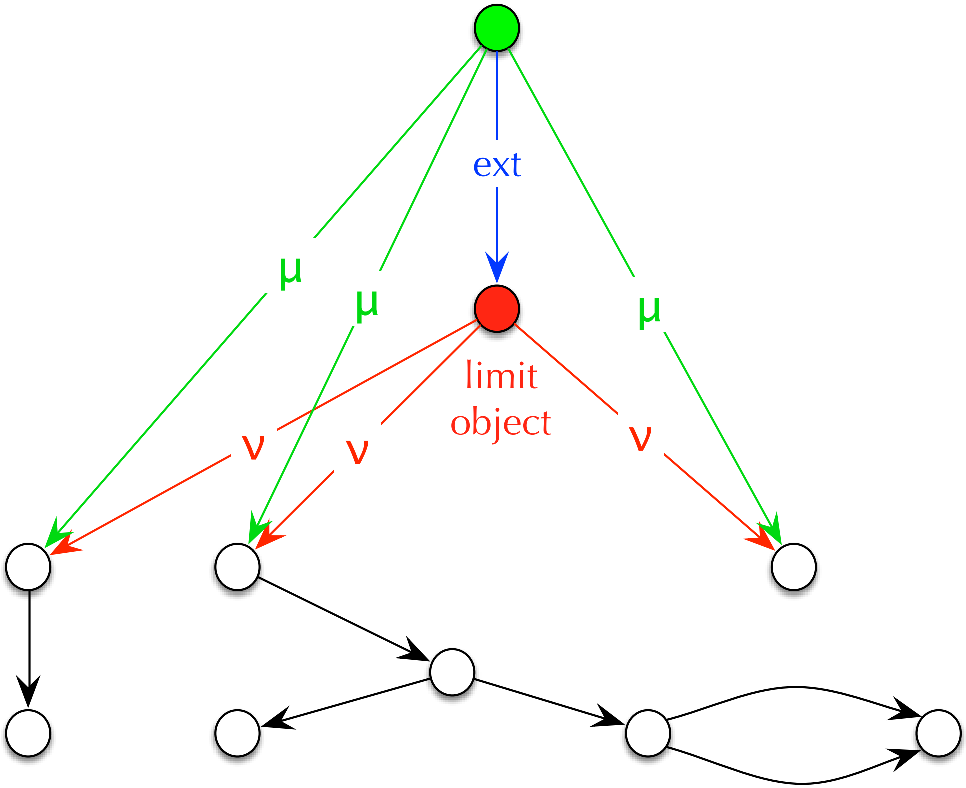
6 Limits and colimits

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

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.

6.1 Limits



A diagram, its limit and a further cone

(Every cone arrow that would be equal to the composition of printed morphisms is omitted.)

A tuple $\mu = (\mu_n : C \to \mathcal{D}(n))_{n\in\mathcal{I}}$ 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 μ. A cone is usually abbreviated to its source.

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

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

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

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

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

$\mathbb{O}$ denotes the category with ordinal numbers as objects and all pairs $(i,j) \in \mathbb{O}^2$ with $i \le j$ as morphisms.

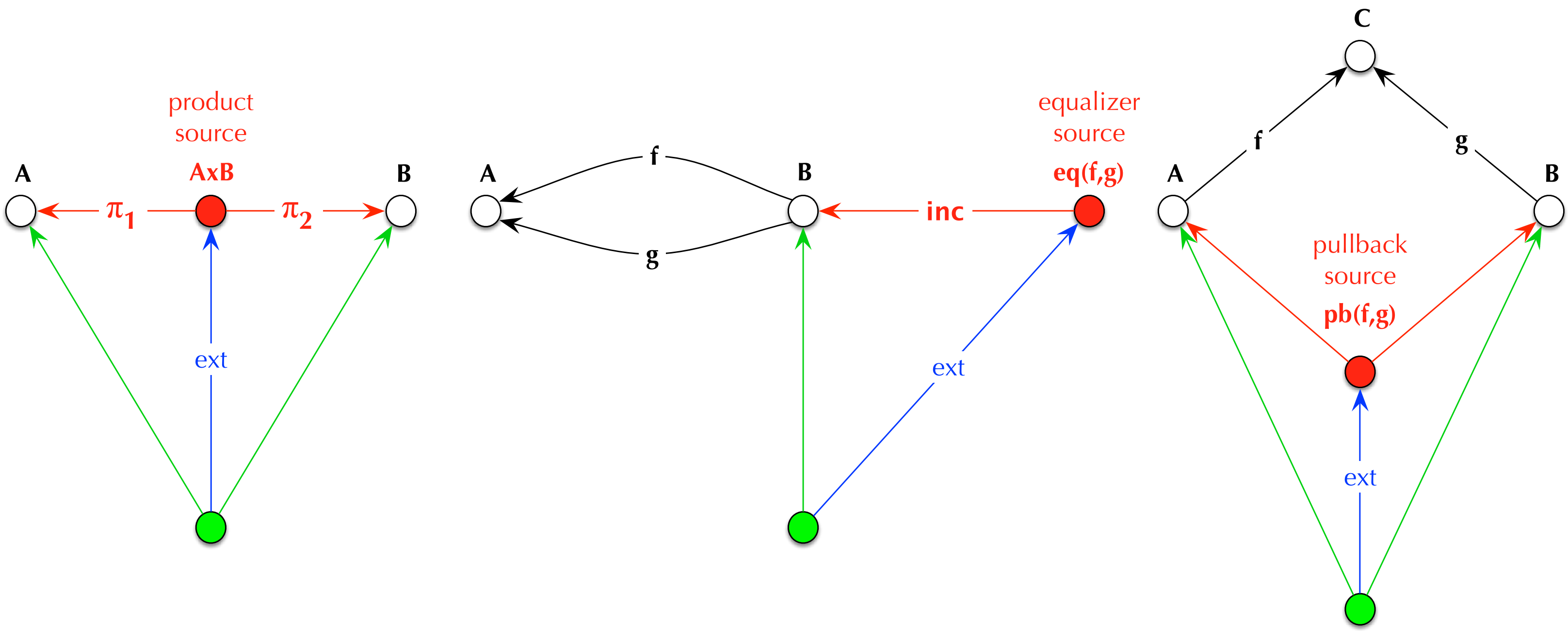
$\mathbb{O}_\lambda$ denotes the full subcategory of $\mathbb{O}$ with all ordinal numbers less than λ as objects.

A **chain** of $\mathcal{K}$ is a diagram $\mathcal{D} : \mathbb{O} \to \mathcal{K}$.

Let λ be an ordinal number. A **λ-chain** of $\mathcal{K}$ is a diagram $\mathcal{D} : \mathbb{O}_\lambda \to \mathcal{K}$. A **$\lambda$-cochain** of $\mathcal{K}$ is a diagram $\mathcal{D} : \mathbb{O}_\lambda \to \mathcal{K}^{op}$.

$\mathcal{K}$ is **λ-complete** if $\mathcal{K}$ has a final object and all λ-cochains of $\mathcal{K}$ have limits.

A functor $F : \mathcal{K} \to \mathcal{L}$ **preserves limits** if for all limits $\mu = (\mu_n : C \to \mathcal{D}(n))_{n\in\mathcal{I}}$ in $\mathcal{K}$, $F(\mu_n) =_{def} (F(\mu_n) : F(C) \to F(\mathcal{D}(n)))_{n\in\mathcal{I}}$ is a limit in $\mathcal{L}$.



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 F be final in $\mathcal{K}$. Then for all $A \in \mathcal{K}$, $A \times F \cong A$.

Let $\mathcal{K} = Set$.

The source of an equalizer of $(f : B \to A,\ g : B \to A)$ 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 $(f : A \to C,\ g : B \to C)$ 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 ν of $\mathcal{D}$ is called a **product** of $\mathcal{D}$ in $\mathcal{K}$, the components of ν are called **projections** and *ext* is called a **product extension**.

Section 2.1 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, by Proposition 2.5, $\mathcal{K}$ has products iff for all nonempty sets I and tuples $(A_i)_{i\in I} \in \mathcal{K}^I$ there are $P \in \mathcal{K}$, $(d_i : P \to A_i)_{i\in I} \in Mor(\mathcal{K})^I$ and a function

$$\langle \_ \rangle_{i\in I} : \mathsf{X}_{i\in I}\mathcal{K}(B, A_i) \;\to\; \mathcal{K}(B, P)$$

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

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

Proposition 6.1 (Russell's paradox)

The product P of all non-empty sets in Set is not in Set:

If P were in Set, then $\mathcal{P}(P)$ were in Set. Hence there would be the surjective projection $\pi : P \to \mathcal{P}(P)$ and thus $p \in P$ with $\pi(p) = A =_{def} \{p \in P \mid p \notin \pi(p)\}$. $p \in A$ would imply $p \notin \pi(p) = A$, while $p \notin A$ would imply $p \in \pi(p) = A$. ↯ ❑

Similar results can be found in section 19.10.

Theorem 6.2 (Subset Theorem; construction of limits in *Set*)

Let $\mathcal{D} : \mathcal{I} \to Set$ 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

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

is a limit of $\mathcal{D}$. ❏

For instance, Theorem 6.2 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

$$(R' \xrightarrow{inc_{R'}} B \times A \xrightarrow{\pi_1} B,\; R' \xrightarrow{inc_{R'}} B \times A \xrightarrow{\pi_2} A)$$

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

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

of $(f : B \to A,\; g : B \to A)$ 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

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

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

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

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

Theorem 6.3 (Limit Theorem; generalizes Theorem 6.2 to complete categories)

Let $\mathcal{K}$ be a complete category, $\mathcal{D} : \mathcal{I} \to \mathcal{K}$ 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 $\{\mathcal{D}(e) \mid e \in \mathcal{I}(m, n)\}$,
- $\prod_{m\in\mathcal{I}} \xrightarrow{ext_1} \prod_{e\in\mathcal{I}(m,n)} \mathcal{D}(n)$ be the product extension of

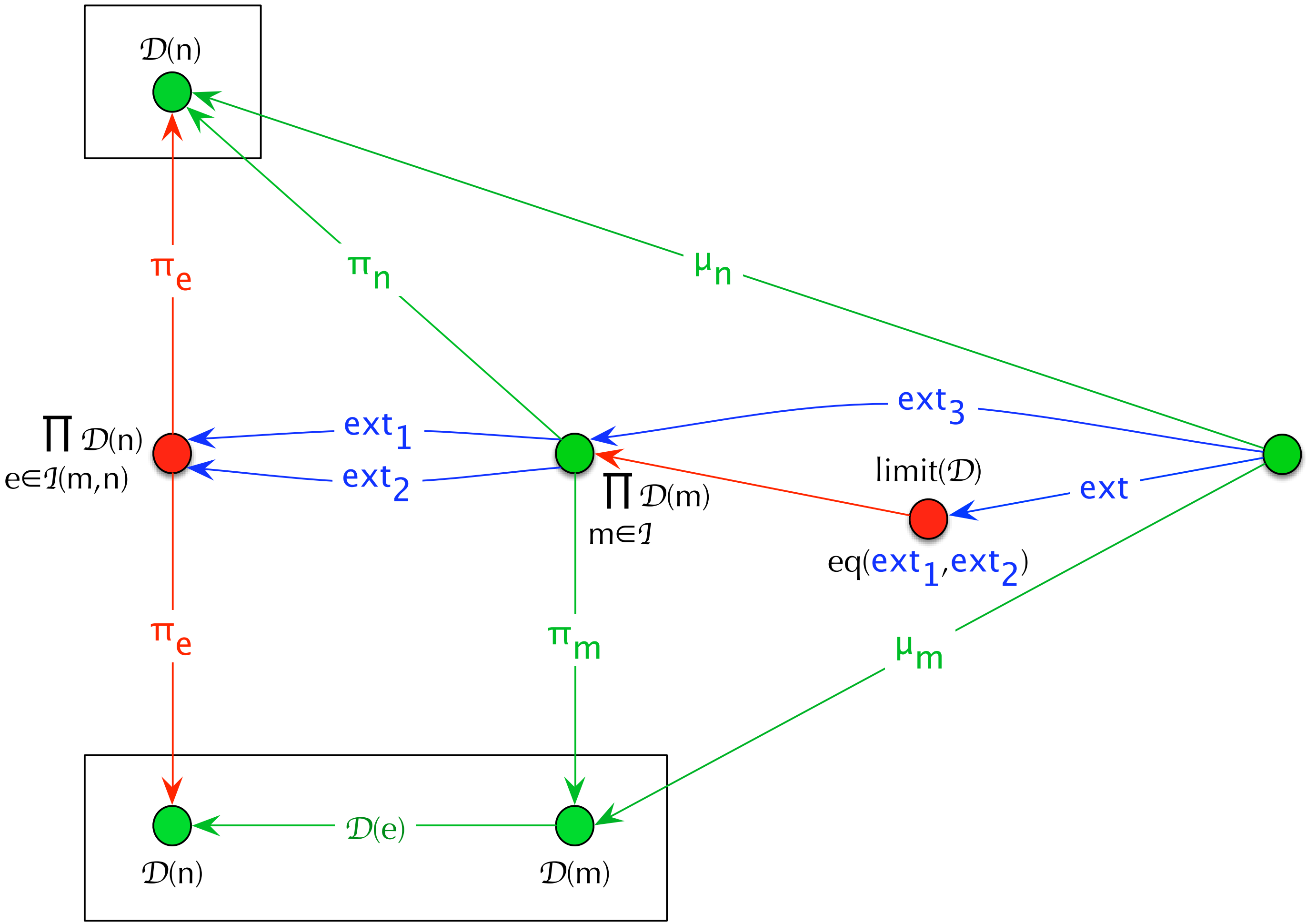
$$(\prod_{m\in\mathcal{I}} \mathcal{D}(m) \xrightarrow{\pi_n} \mathcal{D}(n))_{e\in\mathcal{I}(m,n)},$$

- $\prod_{m\in\mathcal{I}} \xrightarrow{ext_2} \prod_{e\in\mathcal{I}(m,n)} \mathcal{D}(n)$ be the product extension of

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

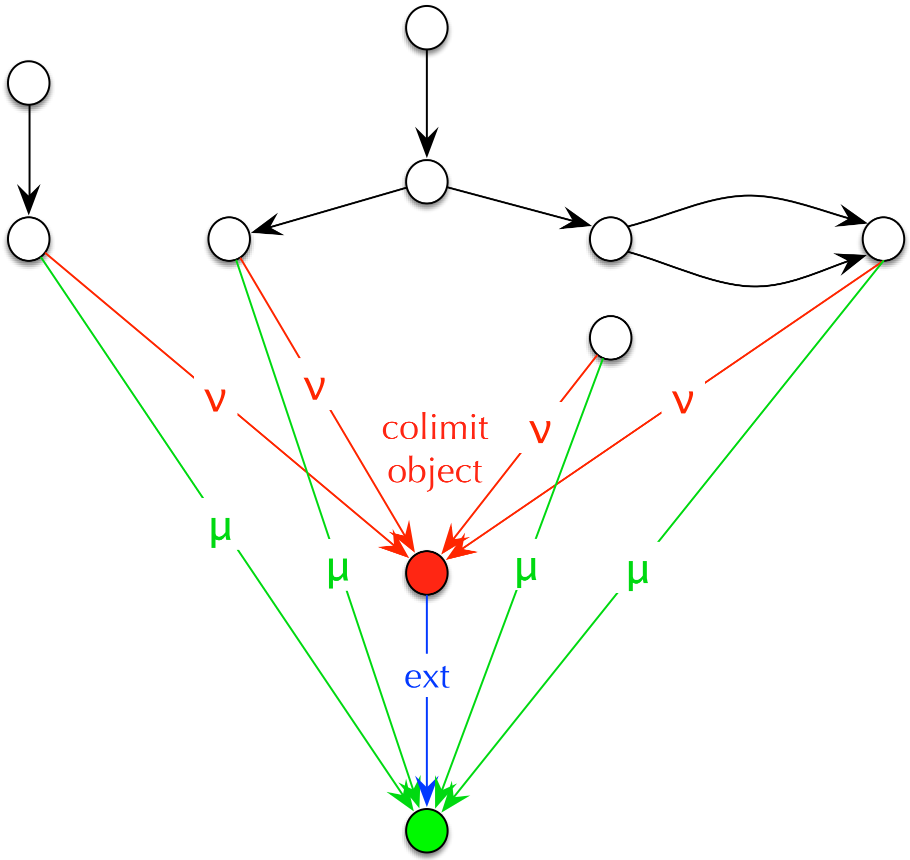
The equalizer of $\{ext_1, ext_2\}$ is a limit of $\mathcal{D}$.

Proof. See the proof of [16], Theorem 2.4.17, or [144], Theorem 1.9.7. ❏



𝒟(n)
π_e
π_n
μ_n
∏ 𝒟(n)
e∈ℐ(m,n)
ext_1
ext_2
ext_3
∏ 𝒟(m)
m∈ℐ
limit(𝒟)
ext
eq(ext_1,ext_2)
π_e
π_m
μ_m
𝒟(e)
𝒟(n)
𝒟(m)

6.2 Colimits



A diagram, its colimit and a further cocone

(Every cocone arrow that would be equal to the composition of printed morphisms is omitted.)

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

A tuple $\mu = (\mu_n : \mathcal{D}(n) \to C)_{n\in\mathcal{I}}$ 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 μ. A cocone is usually abbreviated to its target.

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

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

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

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

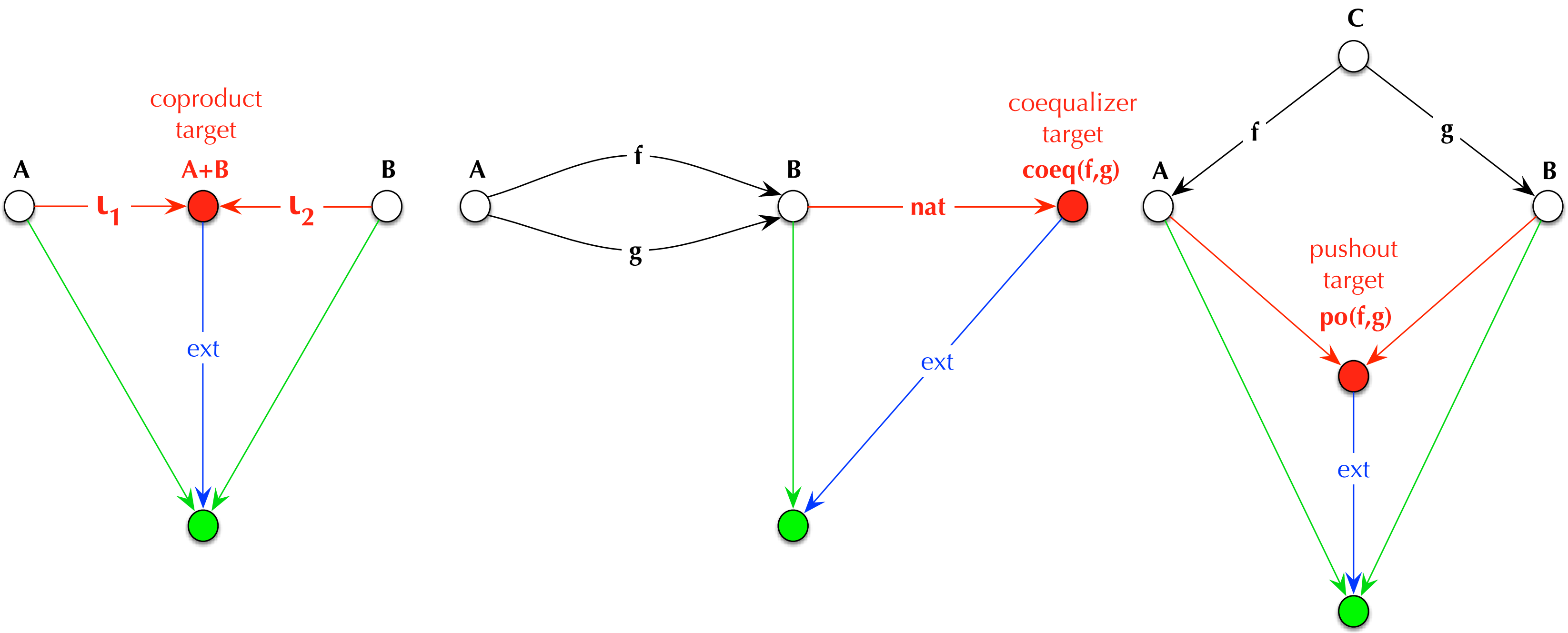
$\mathcal{K}$ is **cocomplete** if each $\mathcal{K}$-diagram has a colimit.

A **cochain** of $\mathcal{K}$ is a diagram $\mathcal{D} : \mathbb{O} \to \mathcal{K}^{op}$.

Let λ be an ordinal number. A **λ-cochain** of $\mathcal{K}$ is a diagram $\mathcal{D} : \mathbb{O}_\lambda \to \mathcal{K}^{op}$.

$\mathcal{K}$ is **λ-cocomplete** if $\mathcal{K}$ has an initial object and all λ-cochains of $\mathcal{K}$ have colimits.

A functor $F : \mathcal{K} \to \mathcal{L}$ **preserves colimits** if for all colimits $\mu = (\mu_n : \mathcal{D}(n) \to C)_{n\in\mathcal{I}}$ in $\mathcal{K}$, $F(\mu_n) =_{def} (F(\mu_n) : F(\mathcal{D}(n)) \to F(C))_{n\in\mathcal{I}}$ is a colimit in $\mathcal{L}$.



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 I be initial in $\mathcal{K}$. Then for all $A \in \mathcal{K}$, $A + I \cong A$.

$\mathcal{K}$ is **distributive** if $\mathcal{K}$ has coproducts and finite products and for all sets I, $A \in \mathcal{K}$ and $(B_i)_{i\in I} \in \mathcal{K}^I$, the unique sum extension

$$[id_A \times \iota_i]_{i\in I} : \coprod_{i\in I}(A \times B_i) \to A \times \coprod_{i\in I} B_i \tag{1}$$

is an isomorphism.

Let $\mathcal{K} = Set$.

Since here sums are disjoint unions (see section 2.4), a binary product $A \times B$ is isomorphic to the sum of A copies of B, i.e., $A \times B \cong \coprod_{a\in A} B$. In particular,

$$\coprod_{i\in I}(A \times B_i) \;\cong\; \coprod_{i\in I}\coprod_{a\in A} B_i \;\cong\; \coprod_{a\in A}\coprod_{i\in I} B_i \;\cong\; A \times \coprod_{i\in I} B_i.$$

The target of a coequalizer of $(f : A \to B,\ g : A \to B)$ 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 $(f : C \to A,\ g : C \to B)$ 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 ν of $\mathcal{D}$ is called a **coproduct** or **sum** of $\mathcal{D}$ in $\mathcal{K}$, the components of ν are called **injections** and *ext* is called a **sum extension**.

Section 2.4 deals with sums in *Set*. All definitions and results presented there—except for the representation of sums by disjoint unions—also apply to $\mathcal{K}$ instead of *Set*.

In particular, by Proposition 2.10, $\mathcal{K}$ has sums iff for all nonempty sets I and tuples $(A_i)_{i \in I} \in \mathcal{K}^I$ there are $S \in \mathcal{K}$, $(c_i : A_i \to S)_{i \in I} \in Mor(\mathcal{K})^I$ and a function

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

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

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

Theorem 6.4 (Quotient Theorem; construction of colimits in Set)

Let $\mathcal{D} : \mathcal{I} \to Set$ 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

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

is a colimit of $\mathcal{D}$. ❏

For instance, Theorem 6.4 provides the following representations of coequalizers and pushouts, respectively:

Let $R' = \{(\iota_1(a), \iota_2(f(a))) \mid a \in A\} \cup \{(\iota_1(a), \iota_2(g(a))) \mid a \in A\}$, $\sim\ = R'^{eq}$ and $S = (A + B)/\sim$. By Theorem 6.4, the cocone

$$(A \xrightarrow{\iota_1} A + B \xrightarrow{nat_\sim} S,\ B \xrightarrow{\iota_2} A + B \xrightarrow{nat_\sim} S)$$

is a colimit of $(f : A \to B,\ g : A \to B)$ and thus S is isomorphic to the coequalizer B/R^{eq} of $(f : A \to B,\ g : A \to B)$ 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 Theorem 6.4, the cocone

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

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

Theorem 6.5 (Colimit Theorem; generalizes Theorem 6.4 to cocomplete categories)

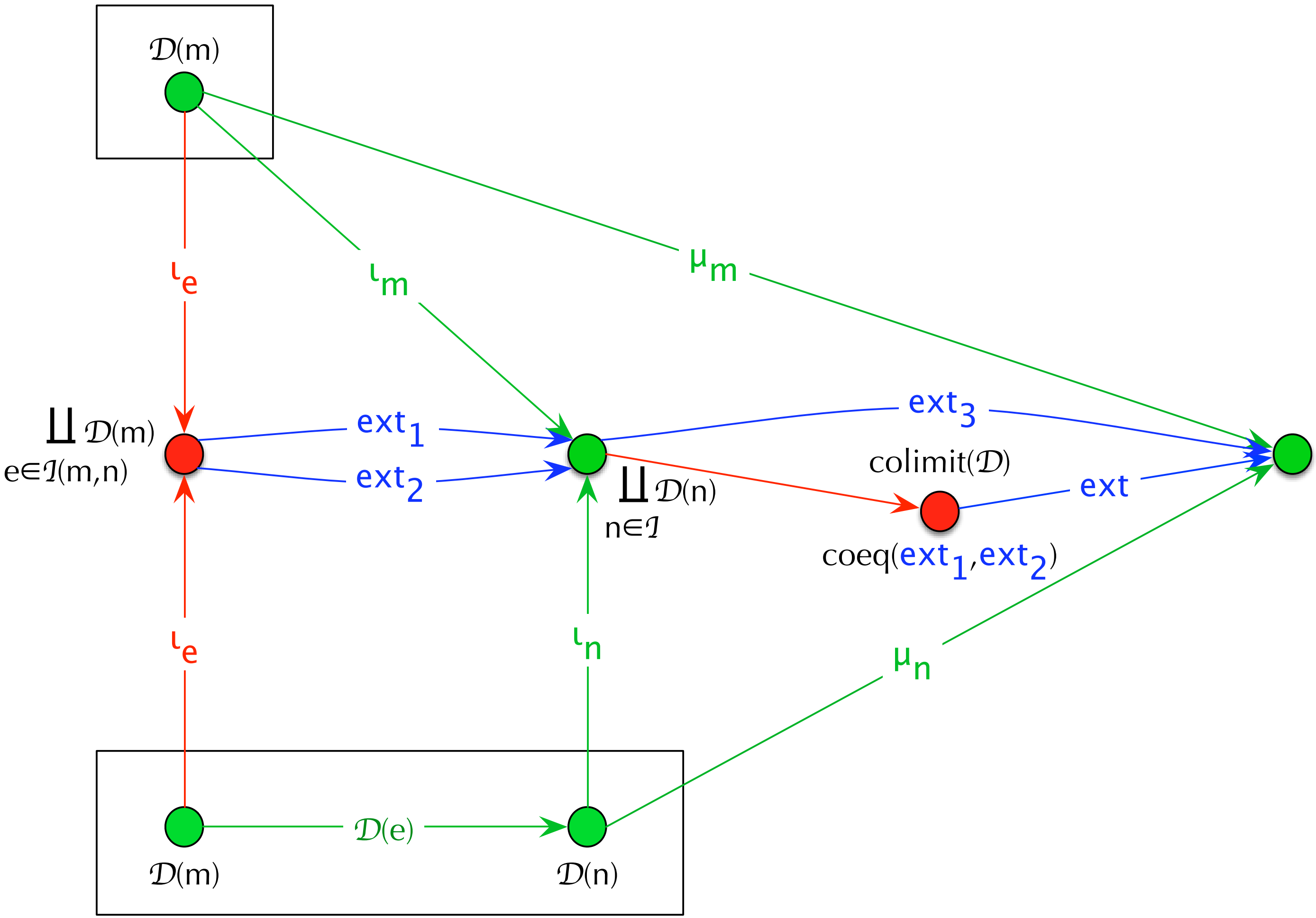
Let $\mathcal{K}$ be a cocomplete category, $\mathcal{D} : \mathcal{I} \to \mathcal{K}$ be a diagram,

- $(\mathcal{D}(n) \xrightarrow{\iota_n} \coprod_{n \in \mathcal{I}} \mathcal{D}(n))_{n \in \mathcal{I}}$ 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 $\{\mathcal{D}(e) \mid e \in \mathcal{I}(m,n)\}$,
- $\coprod_{e \in \mathcal{I}(m,n)} \mathcal{D}(m) \xrightarrow{ext_1} \coprod_{n \in \mathcal{I}} \mathcal{D}(n)$ be the sum extension of
$$(\mathcal{D}(m) \xrightarrow{\iota_m} \coprod_{n \in \mathcal{I}} \mathcal{D}(n))_{e \in \mathcal{I}(m,n)},$$
- $\coprod_{e \in \mathcal{I}(m,n)} \mathcal{D}(m) \xrightarrow{ext_2} \coprod_{n \in \mathcal{I}} \mathcal{D}(n)$ 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 $\{ext_1, ext_2\}$ is a colimit of $\mathcal{D}$.

Proof.

Since $\mathcal{K}^{op}$ is complete, the theorem follows from Theorem 6.3 by dualization. ❑



𝒟(m)
ιe
ιm
μm
∐ 𝒟(m)
e∈𝓘(m,n)
ext1
ext2
ext3
∐ 𝒟(n)
n∈𝓘
colimit(𝒟)
ext
coeq(ext1,ext2)
ιe
ιn
μn
𝒟(m)
𝒟(e)
𝒟(n)

7 Sorted sets and types

Let S be a nonempty set.

The objects of the power category Set^S are tuples $A = (A_s)_{s\in S}$ of sets (see section 4.2) and called S**-sorted** or S**-indexed** sets. $a \in A_s$ is often written as $a : s \in A$. Sometimes A stands for the union of A_s over all $s \in S$.

A morphism $f : A \to B$ in Set^S is a tuple $(f_s : A_s \to B_s)_{s\in S}$ of functions between sets. f is also called an S**-sorted function**.

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.

The objects of the power category Mfn^S are the objects of Set^S. A morphism $f : A \to B$ in Mfn^S is a tuple $(f_s : A_s \to B_s)_{s\in S}$ of multivalued or nondeterministic functions between sets. f is also called a **multivalued** S**-sorted function**.

The S-sorted set A with $A_s = \emptyset$ for all $s \in S$ is initial in Set^S and initial and final in Mfn^S.

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

For all ordinal numbers λ, Set^S is λ-complete and λ-cocomplete.

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

$$A = A_1 \times \cdots \times A_n =_{def} (A_{1,s} \times \ldots \times A_{n,s})_{s \in S}$$

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

Let $\mathcal{I}$ be a fixed set that includes the elements of 1, 2 and $\mathbb{N}$ (see chapter 2).

The subsets of $\mathcal{I}$ provide both **constant types**, also called parameter or primitive types in the sense of, e.g., [190, 191, 27], and index sets of sum and product types.

The sets $\mathcal{T}_s(S)$, $\mathcal{T}_p(S)$, $\mathcal{T}_{po}(S)$, $\mathcal{T}_{fo}(S)$ and $\mathcal{T}(S)$ of **sum**, **product**, **polynomial**, **first-order** and all **types over** S, respectively, are inductively defined as follows:

- $S \cup \mathcal{P}(\mathcal{I}) \subseteq \mathcal{T}_s(S) \cap \mathcal{T}_p(S)$ (sorts and constant types are sum and product types)
- for all $I \subseteq \mathcal{I}$ and $(e_i)_{i \in I} \in \mathcal{T}_{po}(S)^I$, (composed sum and product types)

$$\coprod_{i \in I} e_i \in \mathcal{T}_s(S) \quad \text{and} \quad \prod_{i \in I} e_i \in \mathcal{T}_p(S),$$

- $\mathcal{T}_s(S) \cup \mathcal{T}_p(S) \subseteq \mathcal{T}_{po}(S)$, (product and sum types are polynomial)
- for all $e \in \mathcal{T}_{po}(S)$, commutative monoids $(M, +, 0)$ and $C \subseteq M$, (weighted types)

$$(e \times M)^*_C \in \mathcal{T}_{po}(S) \quad \text{and} \quad M^e_C \in \mathcal{T}_{fo}(S),$$

- $\mathcal{T}_{po}(S) \subseteq \mathcal{T}_{fo}(S)$, (polynomial types are first-order)
- $\mathcal{T}_{fo}(S) \subseteq \mathcal{T}(S)$,
- for all $e, e' \in \mathcal{T}(S)$, $\mathcal{P}(e), e \to e' \in \mathcal{T}(S)$. (powerset and arrow types)

A type is **monomorphic** if it does not contain sorts. Hence $\mathcal{T}(\emptyset)$ is the set of monomorphic 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 $\prod_{i \in I} e_i$.

For all $e, e' \in \mathcal{T}(S)$, $\mathcal{P}(e)$ is called a **relational type** and $e \to e'$ is called a **functional type**. Weighted types are also called **quantitative types**.

Given $e \in \mathcal{T}_{po}(S)$ and $(e_s)_{s \in S} \in \mathcal{T}(S)^S$,

$$e[e_s/s \mid s \in S]$$

denotes the type obtained from e by replacing all occurrences of $s \in S$ with e_s.

Arrow types $e_1 \to (e_2 \to \dots (e_n \to e))$ are often written without brackets:

$$e_1 \to e_2 \to \dots e_n \to e.$$

On the one hand, types over S generalize classical *regular expressions* over $\mathcal{I}$ by admitting sums and products with indices taken from constant types. On the other hand, types over S 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 $\coprod_{i\in e} e_i$ of $\mathcal{T}(S)$ corresponds to a *dependent pair type* and is written as $\sum i:e.e_i$, while $\prod_{i\in I} e_i$ resembles a *dependent function type* and is written as $\prod i:e.e_i$ (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 $I \subseteq \mathcal{I}$, $n > 0$, $e, e_1, \dots, e_n \in \mathcal{T}(S)$ and commutative monoids $(M, +, 0)$,

$$e^I = \prod_{i\in I} e, \qquad \text{(power types)}$$

$$e_1 + \cdots + e_n = \coprod_{i=1}^{n} e_i = \coprod_{i\in[n]} e_i, \qquad \text{(finite sums)}$$

$$
\begin{aligned}
e_1 \times \cdots \times e_n &= \prod_{i=1}^{n} e_1 = \prod_{i \in [n]} e_i, && \text{(finite products)}\\
e^0 &= 1, \\
e^n &= e^{[n]}, \\
e^* &= \coprod_{n \in \mathbb{N}} e^n, && \text{(finite lists)}\\
e^+ &= \coprod_{n > 0} e^n, && \text{(nonempty finite lists)}\\
e^\infty &= e^* + e^{\mathbb{N}}, && \text{(finite or infinite lists)}\\
\mathcal{D}(e) &= \mathbb{R}^{e}_{\geq 0, \{1\}} && \text{(probabilistic types)}
\end{aligned}
$$

7.1 Type models

A $\mathcal{T}(S)$-sorted set $A = (A_e)_{e \in \mathcal{T}_h(S)}$ is a **type model over** S if

- for all $I \subseteq \mathcal{I}$, $A_I = I$,
- for all $I \subseteq \mathcal{I}$ and $(e_i)_{i \in I} \in \mathcal{T}(S)^I$ there are a tuple π of projections (see section 2.1) and a tuple ι of injections (see section 2.4) such that $(A_{\prod_{i \in e} e_i}, \pi)$ is a product and $(A_{\coprod_{i \in e} e_i}, \iota)$ is a sum of $(A_{e_i})_{i \in e}$,

- for all $e \in \mathcal{T}(S)$, commutative monoids M and $C \subseteq M$,

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

(see section 2.8) and

$$A_{(M\times e)_C^*} = \{x \in (M \times A_e)^* \mid \sum_{i=1}^{|x|} map(\pi_1 \circ \pi_i)(x) \in C\},$$

- for all $e, e' \in \mathcal{T}(S)$,

$$A_{\mathcal{P}(e)} = \mathcal{P}(A_e) \quad \text{and} \quad A_{e \to e'} = (A_e \to A_{e'}).$$

Even if a product $P = \prod_{i \in I} A_{e_i}$ is not Cartesian, we sometimes use the tuple notation for elements of P, but keep in mind that it is an abbreviation: For instance, $(a_1, \ldots, a_n)$ stands for $\langle \lambda z.a_1, \ldots, \lambda z.a_n \rangle()$ where z is a variable of type 1.

$Mod(S)$ denotes the class of type models of S.

Two types $e, e' \in \mathcal{T}(S)$ are **equivalent**, written as $e \equiv e'$, if for all type models A of S, $A_e \cong A_{e'}$.

Example 7.1 For all $I \subseteq \mathcal{I}$, $e, e' \in \mathcal{T}(S)$ and $(e_i)_{i\in I} \in \mathcal{T}(S)^I$,

$$I \to e \;\equiv\; e^I, \quad \mathcal{P}(e) \;\equiv\; e \to 2, \quad \mathcal{P}(e \to e') \;\equiv\; e \to \mathcal{P}(e'),$$
$$e \to \mathcal{P}(\textstyle\coprod_{i\in I} e_i) \;\equiv\; \textstyle\prod_{i\in I}(e \to \mathcal{P}(e_i)). \tag{2}$$

Proof of (2). The function h defined by

$$\begin{aligned} \pi_i \circ h : (A \to \mathcal{P}(\textstyle\coprod_{i\in I} A_i)) &\to (A \to \mathcal{P}(A_i)) \qquad i \in I \\ f &\mapsto \lambda a.\{b \in A_i \mid \iota_i(b) \in f(a)\} \end{aligned}$$

has the inverse

$$\begin{aligned} h^{-1} : \textstyle\prod_{i\in I}(A \to \mathcal{P}(A_i)) &\to (A \to \mathcal{P}(\textstyle\coprod_{i\in I} A_i)) \\ g &\mapsto \lambda a.\{\iota_i(b) \in \textstyle\coprod_{i\in I} A_i \mid b \in \pi_i(g)(a),\ i \in I\}. \end{aligned}$$

However, if $\mathcal{P}$ is replaced by the list functor (and lists are represented as usually), the list counterpart h' of h, defined by

$$\begin{aligned} \pi_i \circ h' : (A \to (\textstyle\coprod_{i\in I} A_i)^*) &\to (A \to A_i^*) \qquad i \in I \\ f &\mapsto \lambda a.\mathit{filter}(\lambda\iota_j(b).i = j)(f(a)), \end{aligned}$$

is surjective, but not injective and thus $\coprod_{i\in I}(e \to e_i^*)$ is not equivalent to $e \to (\coprod_{i\in I} e_i)^*$, but to the quotient $(e \to (\coprod_{i\in I} e_i)^*)/\mathit{ker}(h')$ (see chapters 2 and 3). ❑

$Mod(S)$ becomes a subcategory of the power category Set^S (see section 4.2) if we add all S-sorted functions $h : A \to B$ as morphisms, but extend these only to *first-order* types inductively as follows:

- For all $I \subseteq \mathcal{I}$, $h_I = id_I$.
- For all $I \subseteq \mathcal{I}$ and $(e_i)_{i\in I} \in \mathcal{T}_{fo}(S)^e$, $h_{\coprod_{i\in I} e_i} = \coprod_{i\in I} h_{e_i}$ and $h_{\prod_{i\in I} e_i} = \prod_{i\in I} h_{e_i}$.
- For all $e, e' \in \mathcal{T}_{fo}(S)$, commutative monoids M and $C \subseteq M$, $h_{M_C^e} = M_C^{h_e}$ and $h_{(e\times M)^*_M} = (id_M \times h_e)^*$ (see sections 2.8, 2.1 and 2.7).

For each $e \in \mathcal{T}_{fo}(S)$, the "projection" functor

$$F_e : Mod(S) \to Set$$

maps every model A of $\mathcal{T}_{fo}(S)$ to A_e and every $Mod(S)$-morphism h to h_e.

The functor property implies that for all $e \in \mathcal{T}_{fo}(S)$, h_e is a function from A_e to B_e.

A stronger notion of type equivalence than the above one would be the following one:

Two weighted types e, e' over S are equivalent iff F_e and $F_{e'}$ are naturally equivalent (see chapter 5).

7.2 Sorted relations

Let $A_1, \ldots, A_n$ be models of $\mathcal{T}_{fo}(S)$ and $A = A_1 \times \cdots \times A_n$ be their 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 $s \in S$,

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

If $n = 1$, then R is also called an **S-sorted subset of A**.

An S-sorted n-ary relation R on A is lifted to a $\mathcal{T}_{fo}(S)$-sorted n-ary relation as follows:

- For all $I \subseteq \mathcal{I}$, $R_I = \Delta_I^{[n]}$ (see chapter 2).
- For all $I \subseteq \mathcal{I}$ and $(e_i)_{i \in I} \in \mathcal{T}_{fo}(S)^I$,

$$R_{\coprod_{i \in I} e_i} \;=\; \{(\iota_i(a_j))_{j=1}^n \in \times_{j=1}^n A_{j,e} \mid (a_j)_{j=1}^n \in R_{e_i},\; i \in I\}, \tag{3}$$

$$R_{\prod_{i \in I} e_i} = \{(a_k)_{j=1}^n \in \times_{j=1}^n A_{j,e} \mid \forall\, i \in I : (\pi_i(a_j))_{j=1}^n \in R_{e_i}\}. \tag{4}$$

- For all $e, e' \in \mathcal{T}_{fo}(S)$, commutative monoids M and $C \subseteq M$,

$$R_{M_C^e} \;=\; \{(f_1, \ldots, f_n) \in \times_{j=1}^n M_C^{A_j} \mid \exists\, f \in M_C^{R_e} : \forall\, j \in [n] : f_j = \lambda a_j . \textstyle\sum\{f(a) \mid a \in R_e,\; \pi_j(a) = a_j\}, \tag{5}$$

$$R_{(M \times e)_C^*} \;=\; \{((m_{ij}, a_{ij})_{i=1}^k)_{j=1}^n \in \times_{j=1}^n A_{j,e} \mid \forall\, i \in [k] : (a_{i1}, \ldots, a_{in}) \in R_e,\; \forall\, j \in [n] : \textstyle\sum_{i=1}^k m_{ij} \in C\}$$

(see section 2.8).

Exercise 9

Show by induction on e that for all $e \in \mathcal{T}_{fo}(S)$, R_e is a subset of $\bigtimes_{i=1}^{n} A_{e,i}$, and thus R is indeed a $\mathcal{T}_{fo}(S)$-sorted n-ary relation on the extension of A to a $\mathcal{T}_{fo}(S)$-sorted product. ❏

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 chapter 2).

The above-defined lifting of S-sorted to $\mathcal{T}_{fo}(S)$-sorted relations transfers images, kernels and other algebraic relations from Set^S to $Mod(S)$ (see sections 9.9 and 9.10, respectively).

Lemma 7.2 (F_e preserves equalizers and coequalizers)

Let $g, h : A \to B$ be $Mod(S)$-morphisms, 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 $e \in \mathcal{T}_{fo}(S)$,

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

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

Proof. Induction on the size of e. ❏

8 Signatures

For all $e, e' \in \mathcal{T}(S)$, a typed symbol of the form $f : e \to e'$ is called an **arrow** with **source** $src(f) = e$ and **target** $trg(f) = e'$.

$f : e \to e'$ is **finitary** if $e = s_1 \times \cdots \times s_n$ for some $n > 0$ and $s_1, \ldots, s_n \in S$.

If (e is polynomial and) $e' \in S$, then $f : e \to e'$ is a (**polynomial**) e'**-constructor**.

If $e \in S$ (and e' is polynomial), then $f : e \to e'$ is a (**polynomial**) e**-destructor**.

A **signature** $\Sigma = (S, F)$ consists of a set S of **sorts** and a set F of arrows.

Σ is **finitary** if all arrows of F are finite.

Σ is **polynomial** if the sources and targets of all arrows of F are polynomial.

Σ is **constructive** (**destructive**) if all arrows of F are constructors (destructors).

Let $\Sigma = (S, F)$ and $\Sigma' = (S', F')$.

Σ is a **subsignature** of Σ' and Σ' is an **extension** of Σ if Σ' includes Σ componentwise.

A **signature morphism** $\sigma : \Sigma \to \Sigma'$ maps S to $S' \cup \mathcal{P}(\mathcal{I})$ and F to F' such that for all $f : e \to e' \in F$, $\sigma(f) : \sigma^*(e) \to \sigma^*(e') \in F'$ where $\sigma^*(e)$ denotes the type obtained from e by replacing every $s \in S$ with $\sigma(s)$.

The image signature $\sigma(\Sigma)$ is also written as $\Sigma[\sigma(s)/s \mid s \in S,\ \sigma(s) \neq s]$.

$\Sigma \cup \Sigma'$ denotes $(S \cup S', F \cup F')$.

8.1 Σ-arrows

Let $\Sigma = (S, F)$ be a signature. The set Arr_Σ of **Σ-arrows** is the least $\mathcal{T}(S)$-sorted set that contains F and satisfies the following conditions:

- All functions between monomorphic types are Σ-arrows. (monomorphic arrows)
- For all $e \in \mathcal{T}(S)$, $id_e : e \to e \in Arr_\Sigma$. (identities)
- For all $f : e \to e',\ g : e' \to e'' \in Arr_\Sigma$, $g \circ f : e \to e'' \in Arr_\Sigma$. (composition)
- For all $i \in I$, $\iota_i : e_i \to \coprod_{i \in I} e_i \in Arr_\Sigma$. (injections)
- For all $(f_i : e_i \to e)_{i \in I} \in Arr_\Sigma^I$, $[f_i]_{i \in I} : \coprod_{i \in I} e_i \to e \in Arr_\Sigma$. (sum extensions)
- For all $i \in I$, $\pi_i : \prod_{i \in I} e_i \to e_i$. (projections)

- For all $(f_i : e \to e_i)_{i\in I} \in Arr^I_\Sigma$, $\langle f_i\rangle_{i\in I} : e \to \prod_{i\in I} e_i \in Arr_\Sigma$. (product extensions)
- For all $I \subseteq \mathcal{I}$ and $e \in \mathcal{T}(S)$, $get : e^I \times I \to e \in Arr_\Sigma$. (polynomial application)
- For all $f : e \to e' \in Arr_\Sigma$, commutative monoids M and $C \subseteq M$,
  $M^f_C : M^e_C \to M^{e'}_C$, $(M \times f)_C : (e \times M)^*_C \to (e' \times M)^*_C \in Arr_\Sigma$.
  (weighted extensions)
- For all $e, e' \in \mathcal{T}_{fo}(S)$ and natural transformations $\tau : F_e \to F_{e'}$, $\overline{\tau} : e \to e' \in Arr_\Sigma$.
  (type transformations)

  For instance, given $e \in \mathcal{T}_{fo}(S)$, the type transformation

$$\overline{fork_e} : e \times 2 \to e + e$$

  stems from the natural transformation $fork_e : F_{e\times 2} \to F_{e+e}$ that is defined as follows: For all $A \in Mod(S)$, $a \in A_e \times 2$,

$$fork_{e,A}(a) \; = \; \begin{cases} \iota_1(\pi_1(a)) & \text{if } \pi_2(a) = 1, \\ \iota_2(\pi_1(a)) & \text{otherwise.} \end{cases}$$

$g \in Arr_\Sigma$ is a **subarrow** of a $f \in Arr_\Sigma$ if $f = g$ or

- $f \in \{h \circ f', f' \circ h\}$ for some $h \in Arr_\Sigma$ and g is a subarrow of f', or
- $f \in \{[f_i]_{i \in I}, \langle f_i \rangle_{i \in I}\}$ for some $(f_i)_{i \in I} \in Arr_\Sigma^I$ and g is a subarrow of f_i for some $i \in I$.

Derived Σ-arrows

- For all $(f_i : e_i \to e'_i)_{i \in I} \in Arr_\Sigma^I$, (sums and products)

$$\coprod_{i \in I} f_i = [\iota_i \circ f_i]_{i \in I} : \coprod_{i \in I} e_i \to \coprod_{i \in I} e'_i \quad \text{and} \quad \prod_{i \in I} f_i = \langle f_i \circ \pi_i \rangle_{i \in I} : \prod_{i \in I} e_i \to \prod_{i \in I} e'_i.$$

- For all $e, e' \in \mathcal{T}_{fo}(S)$, $p : e \to 2$, $f, g : e \to e' \in Arr_\Sigma$, (tests and conditionals)

$$p? = \overline{fork_e} \circ \langle id_e, p \rangle : e \to e + e \quad \text{and} \quad ite(p, f, g) = [f, g] \circ p? : e \to e'$$

(see [30], section 3.4).

In the following examples, $\leadsto$ points to the carrier of a "standard model", usually given by the carrier of an initial or final Σ-algebra (see chapter 9). For the definitions of sets of labelled trees, see section 2.9.

Many of these examples and those presented in chapters 9, 15, 17 or 24 have been implemented and tested in Haskell (see the modules Coalg.hs and Compiler.hs).

8.2 Sample constructive signatures

Let $X, Y, Act \subseteq \mathcal{I}$ and $(M, +, 0)$ be a commutative monoid.

- Mon (nonempty unlabelled binary trees; e.g., monoids)

$$S = \{mon\}, \quad F = \{one : 1 \to mon,\ mul : mon \times mon \to mon\}.$$

- $Nat \rightsquigarrow \mathbb{N}$

$$S = \{nat\}, \quad F = \{zero : 1 \to nat,\ succ : nat \to nat\}.$$

- $Dyn(X, Y) \rightsquigarrow X^* \times Y$ (dynamics; Y-pointed automata)

$$S = \{state\}, \quad F = \{cons : X \times state \to state,\ \alpha : Y \to state\}.$$

  $List(X) =_{def} Dyn(X, 1) \rightsquigarrow X^*$

  $List(X)$ is equivalent to $(\{state\}, \{list : X^* \to state\})$ (see chapter 15).

  $List(1)$ is equivalent to Nat.

$Nelist(X) =_{def} Dyn(X, X)$. ⇝ X^+

$Nelist(X)$ is equivalent to $(\{state\}, \{list : X^+ \to state\})$.

- $WDyn(X, Y, M)$ (pointed M-weighted automata)

$$S = \{state\}, \quad F = \{cons : X \times state \to M_M^{state},\ \alpha : Y \to state\}.$$

- $coStream(X)$ ⇝ $X^{\mathbb{N}}$ (semiautomata)

$$S = \{state\}, \quad F = \{cons : X \times state \to state\}.$$

- $WcoStream(X, M)$ (M-weighted semiautomata)

$$S = \{state\}, \quad F = \{cons : X \times state \to M_M^{state}\}.$$

- $coDAut(X, Y)$ ⇝ Y^{X^*} (Moore automata with input from X and output from Y)

$$S = \{state\}, \quad F = \{new : state^X \times Y \to state\}.$$

- $coDAut(X, 2)$ ⇝ $\mathcal{P}(X^*)$ (deterministic acceptors of words over X)

- $Bintree(X)$ ⇝ $ftr(2, X)$ (binary trees of finite depth with node labels from X)

$$S = \{btree\}, \quad F = \{\ bjoin : X \times btree \times btree \to btree,\ empty : 1 \to btree\ \}.$$

- $Nebintree(X)$ ⇝ $ftr(2, X) \setminus \{\Omega\}$
(nonempty binary trees of finite depth with node labels from X)

$$S = \{btree\}, \quad F = \{\ bjoin : X \times btree \times btree \to btree,\ ljoin, rjoin : X \times btree \to btree,\ leaf : X \to btree\ \}.$$

- $Nebintree2(X)$ ⇝ $\{t \in ftr(2, X)\setminus\{\Omega\} \mid \forall\, w \in 2^* : w0, w1 \in def(t) \vee w0, w1 \notin def(t)\}$
(nonempty binary trees of finite depth with node labels from X and outdegree 0 or 2)

$$S = \{btree\}, \quad F = \{\ bjoin : X \times btree \times btree \to btree,\ leaf : X \to btree\ \}.$$

- $Tree(X)$ ⇝ $otr(\mathbb{N},X) \cap ftr(\mathbb{N},X)$ (finitely branching trees of finite depth with node labels from X)

$$S = \{tree, trees\}, \quad F = \{ \; join : X \times trees \to tree, \; nil : 1 \to trees, \; cons : tree \times trees \to trees\}.$$

- $Tree_\omega(X)$ ⇝ $otr(\mathbb{N},X) \cap wtr(\mathbb{N},X)$ (finitely or infinitely branching trees of finite depth with node labels from X)

$$S = \{tree\}, \quad F = \{join : X \times tree^\infty \to tree\}.$$

- $ETree(X,Y)$ ⇝ $ftr(Y,X)$ (finitely branching trees of finite depth with node labels from X and edge labels from Y)

$$S = \{tree\}, \quad F = \{join : X \times (Y \times tree)^* \to tree\}.$$

- $Reg(X)$ ⇝ regular expressions over X

$$
\begin{aligned}
S &= \{\ state\ \}, \\
F &= \{\ par : state \times state \to state, && \text{(parallel composition)} \\
&\quad\ seq : state \times state \to state, && \text{(sequential composition)} \\
&\quad\ star : state \to state, && \text{(iteration)} \\
&\quad\ \underline{\ \ } : \mathcal{P}_+(X) \to state, && \text{(base languages)} \\
&\quad\ \widehat{\underline{\ \ }} : 2 \to state\ \}. && \text{(“empty set” } \widehat{0} \text{ and “empty word” } \widehat{1}\text{)}
\end{aligned}
$$

- $Proc(Act)$ ⇝ process expressions (see section 26.3)

$$
\begin{aligned}
S &= \{\ proc\ \}, \\
F &= \{\ pre : Act \times proc \to proc, && \text{(prefixing by an action)} \\
&\quad\ cho : proc \times proc \to proc, && \text{(choice)} \\
&\quad\ par : proc \times proc \to proc, && \text{(parallel composition)} \\
&\quad\ res : proc \times Act \to proc, && \text{(restriction)} \\
&\quad\ rel : proc \times Act^{Act} \to proc\ \}. && \text{(relabelling)}
\end{aligned}
$$

8.3 Sample destructive signatures

Let $X, Y, Act \subseteq \mathcal{I}$ and $(M, +, 0)$ be a commutative monoid.

- $coNat \rightsquigarrow \mathbb{N} \cup \{\omega\} \cong ltr(1,1)$ (see chapter 2)

$$S = \{nat\}, \quad F = \{pred : nat \to nat + 1\}.$$

- $Stream(X) \rightsquigarrow X^{\mathbb{N}}$

$$S = \{state\}, \quad F = \{head : state \to X,\ tail : state \to state\}.$$

- $WStream(X, M)$

$$S = \{state\}, \quad F = \{head : state \to X,\ tail : state \to M_M^{state}\}.$$

- $WStream^*(X, M) \rightsquigarrow otr(M \times \mathbb{N}, X)$

$$S = \{state\}, \quad F = \{head : state \to X,\ tail : state \to (state \times M)^*\}.$$

- $coDyn(X, Y) \rightsquigarrow X^* \times Y \cup X^{\mathbb{N}}$ (codynamics)

$$S = \{state\}, \quad F = \{split : state \to X \times state + Y\}.$$

$coList(X) =_{def} coDyn(X, 1) \rightsquigarrow X^* \cup X^{\mathbb{N}}$

$coList(1)$ is equivalent to $coNat$.

$coNelist(X) =_{def} coDyn(X, X) \rightsquigarrow X^+ \cup X^{\mathbb{N}}$

- $infBintree(X) \rightsquigarrow X^{2^*}$ (binary trees of infinite depth with node labels from X)

$$S = \{btree\}, \quad F = \{left, right : btree \to btree,\ root : btree \to X\}.$$

- $coBintree(X) \rightsquigarrow ltr(2, X)$
(binary trees of finite or infinite depth with node labels from X)

$$S = \{btree\}, \quad F = \{split : btree \to X \times btree \times btree + 1\}.$$

- $coNebintree(X) \rightsquigarrow ltr(2, X) \setminus \{\Omega\}$
(nonempty binary trees of finite or infinite depth with node labels from X)

$$\begin{aligned} S &= \{btree\}, \\ F &= \{split : btree \to X \times btree \times btree + X \times btree + X \times btree + X\}. \end{aligned}$$

- $coNebintree2(X)$
  $\Leftrightarrow$ $\{t \in ltr(2, X) \setminus \{\Omega\} \mid \forall\, w \in 2^* : w0, w1 \in def(t) \vee w0, w1 \notin def(t)\}$
  (nonempty binary trees of finite or infinite depth with node labels from X and out-degree 0 or 2)

$$S = \{btree\}, \quad F = \{split : btree \to X \times btree \times btree + X\}.$$

- $infTree(X)$ $\Leftrightarrow$ $otr(\mathbb{N}, X) \cap fbtr(\mathbb{N}, X) \cap itr(\mathbb{N}, X)$
  (finitely branching trees of infinite depth with node labels from X)

$$\begin{aligned} S &= \{\ tree\ \}, \\ F &= \{\ subtrees : tree \to tree^+, \\ &\qquad root : tree \to X\ \}. \end{aligned}$$

- $coTree_\omega(X)$ $\Leftrightarrow$ $otr(\mathbb{N}, X) \cap fbtr(\mathbb{N}, X)$ (finitely branching trees of finite or infinite depth with node labels from X)

$$\begin{aligned} S &= \{\ tree\ \}, \\ F &= \{\ subtrees : tree \to tree^*, \\ &\qquad root : tree \to X\ \}. \end{aligned}$$

- $coTree(X)$ ⊸ $otr(\mathbb{N}, X)$ (finitely or infinitely branching trees of finite or infinite depth with node labels from X)

$$\begin{aligned} S &= \{\ tree,\ trees\ \}, \\ F &= \{\ subtrees : tree \to trees,\ \ root : tree \to X, \\ &\qquad split : trees \to tree \times trees + 1\ \}. \end{aligned}$$

- $coETree(X, Y)$ (finitely or infinitely branching trees of finite or infinite depth with node labels from X and edge labels from Y)

$$\begin{aligned} S &= \{\ tree,\ trees\ \}, \\ F &= \{\ subtrees : tree \to trees,\ \ root : tree \to X, \\ &\qquad split : trees \to Y \times tree \times trees + 1\ \}. \end{aligned}$$

- $Trans(Act)$ (transition trees whose edges are labelled with actions; see section 26.3)

$$S = \{tree\}, \quad F = \{denode : tree \to (Act \times tree)^*\}.$$

- $Med(X)$ (deterministic Medvedev automata; semiautomata)

$$S = \{state\}, \quad F = \{\delta : state \to state^X\}.$$

$Med(1)$ is equivalent to $coStream(1)$.

- $NMed(X)$ (nondeterministic Medvedev automata)

$$S = \{state\}, \quad F = \{\delta : state \to (2_2^{state})^X\}.$$

- $NMed^*(X) \leadsto otr(X \times \mathbb{N}, 1)$

$$S = \{state\}, \quad F = \{\delta : state \to (state^*)^X\}.$$

- $WMed(X, M)$ (M-weighted Medvedev automata)

$$S = \{state\}, \quad F = \{\delta : state \to (M_M^{state})^X\}.$$

- $WMed^*(X, M) \leadsto otr(X \times M \times \mathbb{N}, 1)$

$$S = \{state\}, \quad F = \{\delta : state \to ((state \times M)^*)^X\}.$$

- $DAut(X, Y) \leadsto Y^{X^*}$ (Moore automata with input from X and output from Y; Y-colored automata)

$$S = \{state\}, \quad F = \{\delta : state \to state^X, \ \beta : state \to Y\}.$$

$DAut(1, Y)$ is equivalent to $Stream(Y)$.

$DAut(2, Y)$ is equivalent to $infBintree(Y)$.

$Acc(X) =_{def} DAut(X, 2) \Leftrightarrow \mathcal{P}(X^*)$ (deterministic acceptors of words over X)

- $Mealy(X, Y) \Leftrightarrow Y^{X^+}$ (Mealy automata)

$$S = \{state\}, \quad F = \{\delta : state \to state^X,\ \beta : state \to Y^X\}.$$

- $PAut(X, Y) \Leftrightarrow ltr(X, Y)$ (partial automata with input from X and output from Y)

$$S = \{state\}, \quad F = \{\delta : state \to (1 + state)^X,\ \beta : state \to Y\}.$$

$PAut(1, Y)$ 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,

$$(1 + A)^2 \times Y \cong (1 + A + A + A^2) \times Y \cong 1 \times Y + A \times Y + A \times Y + A^2 \times Y$$
$$\cong A \times Y \times A + A \times Y + A \times Y + Y.$$

- $NAut(X, Y)$ (nondeterministic automata with input from X and output from Y)

$$S = \{state\},$$
$$F = \{\delta : state \to (2_2^{state})^X,\ \beta : state \to Y\}.$$

$NAcc(X) =_{def} NAut(X,2) \multimap \mathcal{P}(X^*)$ (non-deterministic acceptors of words over X)

$NAut_\times(X,Y)$

$$S = \{state\},$$
$$F = \{\delta : state \to (X \times 2_2^{state}),\ \beta : state \to Y\}.$$

$NAut^*(X,Y) \multimap otr(X \times \mathbb{N}, Y)$

$$S = \{state\},$$
$$F = \{\delta : state \to (state^*)^X,\ \beta : state \to Y\}.$$

$NAut^*_\times(X,Y)$

$$S = \{state\},$$
$$F = \{\delta : state \to (X \times state)^*,\ \beta : state \to Y\}.$$

By (3) and (4) in chapter 8 (for $e = 1$ and $I = X$), $\mathcal{P}(X \times state)$ and $\mathcal{P}(state)^X$ 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)

$$S = \{state\}, \quad F = \{\delta : state \to (M_M^{state})^X,\ \beta : state \to Y\}.$$

$WAut(1,M,Y)$ is equivalent to $WStream(Y,M)$.

- $WAut^*(X,M,Y) \looparrowright otr(X \times M \times \mathbb{N}, Y)$

$$S = \{state\}, \quad F = \{\delta : state \to ((state \times M)^*)^X,\ \beta : state \to Y\}.$$

- $PrAut(X,Y)$ (probabilistic automata with input from X and output from Y)

$$S = \{state\}, \quad F = \{\delta : state \to \mathcal{D}(state)^X,\ \beta : state \to Y\}.$$

- Let $\Sigma = (S,C)$ be a finitary signature (see above).

$TAcc(\Sigma) \looparrowright \mathcal{P}(T_\Sigma)$ (deterministic top-down tree acceptors; see section 9.3)

$$F = \{\delta_c : s \to s'_1 \times \dots \times s'_n \mid c : s_1 \times \dots \times s_n \to s \in C\}$$

where for all $s \in S \cup \mathcal{P}(\mathcal{I})$, $s' =_{def}$ *if* $s \in S$ *then* s *else* $\mathcal{P}(s)$.

If S and $\mathcal{I}$ are singletons, say $S = \{state\}$ and $\mathcal{I} = \{Y\}$, and for all $c : e \to s \in C$, $e \in \{state, Y\}$, then $TAcc(\Sigma)$ is equivalent to $Mealy(C', \mathcal{P}(Y))$ where

$$C' = \{c : e \to s \in C \mid e = state\}.$$

$NTAcc(\Sigma) \multimap \mathcal{P}(T_\Sigma)$ (nondeterministic top-down tree acceptors)

$$F = \{\delta_c : s \to 2_2^e \mid c : e \to s \in C\}.$$

$NTAcc^*(\Sigma)$ (polynomial nondeterministic top-down tree acceptors)

$$F = \{\delta_c : s \to e^* \mid c : e \to s \in C\}.$$

- *KripkeSig* (silent as well as labelled state transitions and atom valuations)

$$\begin{array}{rcl} S &=& \{\ state, label, atom\ \}, \\ F &=& \{\ inits : 1 \to \mathcal{P}(state), \\ && \quad trans : state \to \mathcal{P}(state), \\ && \quad transL : state \to label \to \mathcal{P}(state), \\ && \quad value : atom \to \mathcal{P}(state), \\ && \quad valueL : atom \to label \to \mathcal{P}(state)\ \}. \end{array}$$

- Let $X_1, \ldots, X_n, Y_1, \ldots, Y_n, E_1, \ldots, E_n$ be nonempty sets and

$$BS = \{X_1, \ldots, X_n, Y_1, \ldots, Y_n, E_1, \ldots, E_n\}.$$

$Class(BS)$ (object classes with n methods [95])

$$S = \{state\}, \quad F = \{m_i : state \to ((Y_i \times state) + E_i)^{X_i} \mid 1 \le i \le n\}.$$

- UML diagrams (object class diagrams with n classes and "associations")

$$S = \{s_1, \ldots, s_n\}, \quad F = \{assoc_i : s_i \to s_{k_1} \times \ldots \times s_{k_i} \mid 1 \le i \le n,\ 1 \le k_j \le n\}.$$

- $Graph(X, Y)$ (node- and edge-labelled graphs)

$$S = \{node, edge\},$$
$$F = \{source, target : edge \to node,\ nlabel : node \to X,\ elabel : edge \to Y\}.$$

9 Σ-algebras

9.1 Algebras and homomorphisms

Let $\Sigma = (S, F)$ be a signature.

A **Σ-algebra** $\mathcal{A}$ is a type model A over S, called the **carrier of** $\mathcal{A}$, together with an **interpretation** of each $f : e \to e' \in F$ as a function $f^{\mathcal{A}} : A_e \to A_{e'}$. The interpretation of F is extended to Arr_Σ inductively as follows:

Let M be a commutative monoid and $C \subseteq M$.

- For all monomorphic types e, e' and $f : e \to e' \in Arr_\Sigma$, $f^{\mathcal{A}} = f$.
- For all $e \in \mathcal{T}(S)$, $id_e^{\mathcal{A}} = id_{A_e}$.
- For all $e \in \mathcal{T}(S)$, $a \in A_e$ and $i \in \mathcal{I}$, $\bar{i}^{\mathcal{A}}(a) = i$, $\iota_i^{\mathcal{A}} = \iota_i$ and $\pi_i^{\mathcal{A}} = \pi_i$.
- For all $f : e \to e'$, $g : e' \to e'' \in Arr_\Sigma$, $(g \circ f)^{\mathcal{A}} = g^{\mathcal{A}} \circ f^{\mathcal{A}}$.
- For all $(f_i : e_i \to e)_{i \in I} \in Arr_\Sigma^I$, $[f_i]_{i \in I}^{\mathcal{A}} = [f_i^{\mathcal{A}}]_{i \in I}$.
- For all $(f_i : e \to e_i)_{i \in I} \in Arr_\Sigma^I$, $\langle f_i \rangle^{\mathcal{A}} = \langle f_i^{\mathcal{A}} \rangle_{i \in I}$.
- For all $I \subseteq \mathcal{I}$, $e \in \mathcal{T}(S)$, $a \in A_e^I$ and $i \in I$, $get^{\mathcal{A}}(a, i) = \pi_i(a)$.
- For all $f : e \to e' \in Arr_\Sigma$, $(M_C^f)^{\mathcal{A}} = M_C^{f^{\mathcal{A}}}$ (see section 2.8).
- For all $f : e \to e' \in Arr_\Sigma$, $(M \times f)_C^{\mathcal{A}} = (id_M \times f^{\mathcal{A}})^*$ (see sections 2.1 and 2.7).

- For all $e, e' \in \mathcal{T}_{fo}(S)$ and natural transformations $\tau : F_e \to F_{e'}$, $\overline{\tau}^{\mathcal{A}} = \tau_A$.

See chapter 2 for the functions and operators on the right-hand sides of the above equations.

We often write $\mathcal{A}(e)$ for A_e.

Exercise 10 Show that every algebra $\mathcal{A}$ interprets derived arrows as desired, e.g.,

- for all $(f_i : e_i \to e'_i)_{i \in I} \in Arr^I_\Sigma$,
$$(\textstyle\coprod_{i \in I} f_i)^{\mathcal{A}} = [\iota_i \circ f_i^{\mathcal{A}}]_{i \in I} \quad \text{and} \quad (\textstyle\prod_{i \in I} f_i)^{\mathcal{A}} = \langle f_i^{\mathcal{A}} \circ \pi_i \rangle_{i \in I},$$
- for all $f = (f_s : e_s \to e'_s)_{s \in S} \in Arr^I_\Sigma$ and $(e_i)_{i \in I} \in \mathcal{T}(S)^I$,
$$(f_{\prod_{i \in I} e_i})^{\mathcal{A}} = \textstyle\prod_{i \in I} f_{e_i}^{\mathcal{A}} \quad \text{and} \quad (f_{\coprod_{i \in I} e_i})^{\mathcal{A}} = \textstyle\coprod_{i \in I} f_{e_i}^{\mathcal{A}}. \qquad ❏$$

Given $f, g \in Arr_\Sigma$, $\mathcal{A}$ **satisfies** the **Σ-arrow equation** $f = g$, written as $\mathcal{A} \models f = g$, if $f^{\mathcal{A}} = g^{\mathcal{A}}$.

Exercise 11 The following example stems from [64], one of the first papers on algebraic software specifications. Let $Domain, Range \subseteq \mathcal{I}$, $equal : Domain^2 \to 2$ be the equality on $Domain$ and $\Sigma = (S, F)$ with

$$
\begin{aligned}
S &= \{\ \mathit{Array}\},\\
F &= \{\ \mathit{new} : \mathit{1} \to \mathit{Array},\\
&\qquad \mathit{assign} : \mathit{Array} \times \mathit{Domain} \times \mathit{Range} \to \mathit{Array},\\
&\qquad \mathit{access} : \mathit{Array} \times \mathit{Domain} \to \mathit{Range} + \mathit{1}\ \}.
\end{aligned}
$$

Define a (simple) Σ-algebra $\mathcal{A}$ that satisfies the following arrow equations:

$$
\begin{aligned}
\mathit{access} \circ \langle \mathit{new} \circ \overline{()}, \mathit{id} \rangle &= \iota_2 \circ \overline{()}\\
&\quad : \mathit{Range} + 1,\\
\mathit{access} \circ \langle \mathit{assign} \circ \langle \pi_1, \pi_2, \pi_3 \rangle, \pi_4 \rangle &= \mathit{ite}(\mathit{equal} \circ \langle \pi_2, \pi_4 \rangle, \iota_1 \circ \pi_3, \mathit{access} \circ \langle \pi_1, \pi_4 \rangle)\\
&\quad : \mathit{Range} + 1,\\
\mathit{assign} \circ \langle \mathit{assign} \circ \langle \pi_1, \pi_2, \pi_3 \rangle, \pi_4, \pi_5 \rangle &= \mathit{ite}(\mathit{equal} \circ \langle \pi_2, \pi_4 \rangle, \mathit{assign} \circ \langle \pi_1, \pi_4, \pi_5 \rangle,\\
&\qquad \mathit{assign} \circ \langle \mathit{assign} \circ \langle \pi_1, \pi_4, \pi_5 \rangle, \pi_2, \pi_3 \rangle)\\
&\quad : \mathit{Array}.
\end{aligned}
$$

Σ-arrow equations are not always the most comprehensible way for expressing desired properties of Σ-algebras. Further formulas (including variables and λ-terms) are provided in sections 9.11 and 9.16 and chapter 10. ❑

Ologs [172] and *sketches* [24, 25] are approaches to specify models in terms of signature graphs and to interpret their nodes and edges by sets and functions, respectively. The graphs are regarded as database *schemas* and the sets and functions they are mapped to represent relational databases (tables) and their attributes, respectively.

Olog example

[172], Example 4.5.2.1, defines a database schema as a quotient category that resembles the free category over a signature. For instance, let $\Sigma = (S, F)$ with

$$\begin{array}{lcl} S & = & \{ \ \mathit{Employee}, \mathit{Department}, \mathit{String}\}, \\ F & = & \{ \ \mathit{manager} : \mathit{Employee} \to \mathit{Employee}, \\ & & \quad \mathit{worksIn} : \mathit{Employee} \to \mathit{Department}, \\ & & \quad \mathit{secretary} : \mathit{Department} \to \mathit{Employee}, \\ & & \quad \mathit{first}, \mathit{last} : \mathit{Employee} \to \mathit{String}, \\ & & \quad \mathit{name} : \mathit{Department} \to \mathit{String} \ \}. \end{array}$$

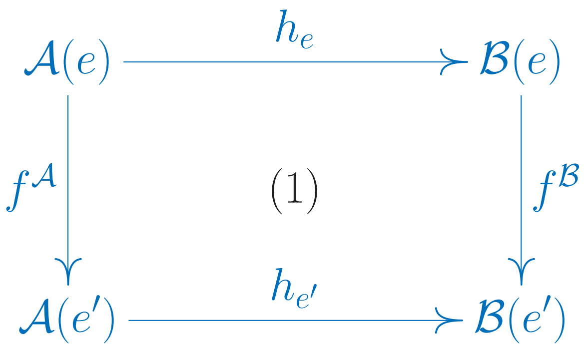
Database constraints are then specified as arrow equations, e.g.,

$$\begin{array}{rcl} \mathit{worksIn} \circ \mathit{manager} & = & \mathit{worksIn} : \mathit{Department}, \\ \mathit{worksIn} \circ \mathit{secretary} & = & id_{\mathit{Department}} : \mathit{Department}. \end{array} \qquad ❏$$

Homomorphisms

Let $\Sigma = (S, F)$ be a signature and $\mathcal{A}, \mathcal{B}$ be Σ-algebras.

A $Mod(S)$-morphism $h : A \to B$ is a **Σ-homomorphism** (or **Σ-homomorphic**) and denoted by $h : \mathcal{A} \to \mathcal{B}$ if for all $e, e' \in \mathcal{T}_{fo}(S)$ and $f : e \to e' \in F$ the following diagram commutes:



h is a **Σ-isomorphism** if h is iso in Alg_Σ.

The category of Σ-algebras and Σ-homomorphisms is a subcategory of $Mod(S)$ and denoted by Alg_Σ.

For all Σ-homomorphisms $h : \mathcal{A} \to \mathcal{B}$,

h is epi in Alg_Σ iff h is epi in $Mod(S)$ iff h is epi in Set^S.

h is mono in Alg_Σ iff h is mono in $Mod(S)$ iff h is mono in Set^S.

h is iso in Alg_Σ iff h is iso in $Mod(S)$ iff h is iso in Set^S.

Lemma 9.1

Let $\mathcal{A}, \mathcal{B}, \mathcal{C}$ be Σ-algebras with carriers A, B, C, respectively, and $g : A \to B$ and $h : B \to C$ be S-sorted functions such that $h \circ g$ is Σ-homomorphic.

(2) If g is epi in Alg_Σ, then h is Σ-homomorphic.

(3) If h is mono in Alg_Σ, then g is Σ-homomorphic.

Proof. Diagram chasing. ❏

Lemma 9.2

Given a Σ-homomorphism $h : \mathcal{A} \to \mathcal{B}$, (1) holds true for all $f : e \to e' \in Arr_\Sigma$ with $e, e' \in \mathcal{T}_{fo}(S)$. In other words, Σ-homomorphisms are also (S, Arr_Σ)-homomorphic.

Proof. We show (1) by induction on the structure of f.

If $f \in F$, then (1) holds true because h is Σ-homomorphic.

If $f = id_e : e \to e$, then

$$h_e \circ f^{\mathcal{A}} = h_e \circ id_{\mathcal{A}(e)} = h_e \circ id_{\mathcal{A}(e)} = h_e \circ id_{\mathcal{B}(e'} = h_e = id_{\mathcal{B}(e)} \circ h_e = f^{\mathcal{B}} \circ h_e.$$

If $f = \bar{i} : e \to I$, then $h_I \circ f^{\mathcal{A}} = id_I \circ f^{\mathcal{A}} = f^{\mathcal{A}} = \lambda a.i = \lambda b.i \circ h_e = f^{\mathcal{B}} \circ h_e$.

If f is an injection, say $f = \iota_i : e_i \to \coprod_{i \in I} e_i$, then

$$h_{e'} \circ f^{\mathcal{A}} = h_{\coprod_{i \in I} e_i} \circ \iota_i = \coprod_{i \in I} h_{e_i} \circ \iota_i \overset{(18)\ in\ section\ 2}{=} \iota_i \circ h_{e_i} = f^{\mathcal{B}} \circ h_e.$$

If f is a projection, say $f = \pi_i : \prod_{i \in I} e_i \to e_i$, then

$$h_{e'} \circ f^{\mathcal{A}} = h_{e_i} \circ \pi_i \overset{(7)\ in\ section\ 2}{=} \pi_i \circ \prod_{i \in I} h_{e_i} = \pi_i \circ h_{\prod_{i \in I} e_i} = f^{\mathcal{B}} \circ h_e.$$

Let $f = f_2 \circ f_1$ for some $f_1 : e \to e''$, $f_2 : e'' \to e' \in Arr_\Sigma$. Then (4) holds true for f_1, f_2 by induction hypothesis. Hence

$$h_{e'} \circ f^{\mathcal{A}} = h_{e'} \circ (f_2 \circ f_1)^{\mathcal{A}} = h_{e'} \circ f_2^{\mathcal{A}} \circ f_1^{\mathcal{A}} \overset{ind.\ hyp.}{=} f_2^{\mathcal{B}} \circ h_{e''} \circ f_1^{\mathcal{A}} \overset{ind.\ hyp.}{=} f_2^{\mathcal{B}} \circ f_1^{\mathcal{B}} \circ h_e$$
$$= (f_2 \circ f_1)^{\mathcal{B}} \circ h_e = f^{\mathcal{B}} \circ h_e.$$

Let $f = [f_i]_{i \in I}$ for some $f_i : e_i \to e'$, $i \in I$. Then $e = \coprod_{i \in I} e_i$ and (1) holds true for f_i, $i \in I$, by induction hypothesis. Hence for all $i \in I$,

$$h_{e'} \circ f^{\mathcal{A}} \circ \iota_i = h_{e'} \circ [f_i]^{\mathcal{A}}_{i\in I} \circ \iota_i = h_{e'} \circ f_i^{\mathcal{A}} \stackrel{ind.\ hyp.}{=} f_i^{\mathcal{B}} \circ h_{e_i} = [f_i^{\mathcal{B}}]_{i\in I} \circ \iota_i \circ h_{e_i}$$
$$\stackrel{(18)\ in\ section\ 2.6}{=} [f_i^{\mathcal{B}}]_{i\in I} \circ \coprod_{i\in I} h_{e_i} \circ \iota_i = [f_i]^{\mathcal{B}}_{i\in I} \circ h_e \circ \iota_i = f^{\mathcal{B}} \circ h_e \circ \iota_i$$

and thus (1) by (13) in section 2.4.

Let $f = \langle f_i \rangle_{i\in I}$ for some $f_i : e \to e_i$, $i \in I$. Then $e' = \prod_{i\in I} e_i$ and (4) holds true for f_i, $i \in I$, by induction hypothesis. Hence for all $i \in I$,

$$\pi_i \circ h_{e'} \circ f^{\mathcal{A}} = \pi_i \circ h_{\prod_{i\in I} e_i} \circ \langle f_i \rangle^{\mathcal{A}}_{i\in I} = \pi_i \circ \prod_{i\in I} h_{e_i} \circ \langle f_i^{\mathcal{A}} \rangle_{i\in I}$$
$$\stackrel{(7)\ in\ section\ 2.3}{=} h_{e_i} \circ \pi_i \circ \langle f_i^{\mathcal{A}} \rangle_{i\in I} = h_{e_i} \circ f_i^{\mathcal{A}} \stackrel{ind.\ hyp.}{=} f_i^{\mathcal{B}} \circ h_e = \pi_i \circ \langle f_i^{\mathcal{B}} \rangle_{i\in I} \circ h_e$$
$$= \pi_i \circ \langle f_i \rangle^{\mathcal{B}}_{i\in I} \circ h_e = \pi_i \circ f^{\mathcal{B}} \circ h_e$$

and thus (1) by (2) in section 2.1.

If $f = get : e \times I \to e$ for some $e \in \mathcal{T}(S)$ and $I \subseteq \mathcal{I}$, then

$$h_e \circ f^{\mathcal{A}} = h_e \circ \lambda(a,i).\pi_i(a) = \lambda(a,i).h_e(\pi_i(a)) = \lambda(a,i).\pi_i(h_e^I(a)) = \lambda(a,i).f^{\mathcal{B}}(h_e^I(a), i)$$
$$= f^{\mathcal{B}} \circ \lambda(a,i).(h_e^I(a), i) = f^{\mathcal{B}} \circ \lambda(a,i).(h_e^I(a), id_I(i)) = f^{\mathcal{B}} \circ \lambda(a,i).(h_e^I(a), h_I(i))$$
$$= f^{\mathcal{B}} \circ \lambda(a,i).h_{e^I\times I}(a,i) = f^{\mathcal{B}} \circ h_{e^I\times I}.$$

If $f = M_C^g : M_C^e \to M_C^{e'}$ for some $g : e \to e' \in Arr_\Sigma$, then for all $f' \in M_C^{A_e}$,

$$h_{M_C^{e'}} \circ f^{\mathcal{A}} = h_{M_C^{e'}} \circ M_C^{g,\mathcal{A}} = h_{M_C^{e'}} \circ M_C^{g^{\mathcal{A}}} = M_C^{h_{e'}} \circ M_C^{g^{\mathcal{A}}} \overset{M_C^{-}\ is\ a\ functor}{=} M_C^{h_{e'} \circ g^{\mathcal{A}}}$$
$$\overset{ind.\ hyp.}{=} M_C^{g^{\mathcal{B}} \circ h_e} \overset{M_C^{-}\ is\ a\ functor}{=} M_C^{g^{\mathcal{B}}} \circ M_C^{h_e} = M_C^{g,\mathcal{B}} \circ h_{M_C^e} = f^{\mathcal{B}} \circ h_{M_C^e}.$$

If $f = (id_M \times g)^*_C : (M \times e)^*_C \to (M \times e')^*_C$ for some $g : e \to e' \in Arr_\Sigma$, then for all $x = ((m_i, a_i))_{i=1^n} \in (M \times A_e)^*_C$,

$$h_{(M\times e')^*_C} \circ f^{\mathcal{A}} = h_{(M\times e')^*_C} \circ ((id_M \times g)^*_C)^{\mathcal{A}} = h_{(M\times e')^*_C} \circ (id_M \times g^{\mathcal{A}})^*_C$$
$$= (id_M \times h_{e'})^*_C \circ (id_M \times g^{\mathcal{A}})^*_C \overset{(id_M\times\_)^*_C\ is\ a\ functor}{=} (id_M \times (h_{e'} \circ g^{\mathcal{A}}))^*_C$$
$$\overset{ind.\ hyp.}{=} (id_M \times (g^{\mathcal{A}} \circ h_e))^*_C \overset{(id_M\times\_)^*_C\ is\ a\ functor}{=} (id_M \times g^{\mathcal{B}})^*_C \circ (id_M \times h_e)^*_C$$
$$= ((M \times g)^*_C)^{\mathcal{B}} \circ h_{(M\times e)^*_C} = f^{\mathcal{B}} \circ h_{(M\times e)^*_C}.$$

If $f = \overline{\tau}$ for some natural transformation $\tau : F_e \to F_{e'}$, then

$$h_{e'} \circ f^{\mathcal{A}} = F_{e'}(h) \circ \tau_A = \tau_B \circ F_e(h) = f^{\mathcal{B}} \circ h_{e'}.$$ ❏

Arr_Σ may be extended by further constants $f : e \to e'$ such that (4) holds true. For instance, every well-founded Σ-term over V (see section 9.3) provides an equivalent Σ-arrow (see section 9.11).

A Σ-homomorphism $h : \mathcal{A} \to \mathcal{B}$ induces the **image algebra** $h(\mathcal{A})$:

- For all $e \in \mathcal{T}_{fo}(S)$, $h(\mathcal{A})(e) = h_e(\mathcal{A}(e))$.
- For all $f : e \to e' \in F$ and $a \in \mathcal{A}(e)$, $f^{h(\mathcal{A})}(h(a)) = f^{\mathcal{B}}(h(a))$.

Reducts

Let $\Sigma = (S, F)$ and $\Sigma' = (S', F')$ be signatures, $\sigma : \Sigma \to \Sigma'$ be a signature morphism and $\mathcal{A}, \mathcal{B}$ be Σ'-algebras.

The **σ-reduct of $\mathcal{A}$**, $\mathcal{A}|_\sigma$, is the Σ-algebra that is defined as follows:

- For all $e \in \mathcal{T}(S)$, $\mathcal{A}|_\sigma(e) = \mathcal{A}(\sigma(e))$.
- For all $f \in F$, $f^{\mathcal{A}|_\sigma} = \sigma(f)^{\mathcal{A}}$.

Let $h : \mathcal{A} \to \mathcal{B}$ be a Σ'-homomorphism.

The **σ-reduct of** h, $h|_\sigma : \mathcal{A}|_\sigma \to \mathcal{B}|_\sigma$, is the Σ-homomorphism that is defined as follows: For all $s \in S$, $(h|_\sigma)_s = h_{\sigma(s)}$.

σ-reducts are the images of the **reduct functor** $\_|_\sigma : Alg_{\Sigma'} \to Alg_\Sigma$.

If σ is an inclusion of signatures, $\mathcal{A}|_\sigma$ and $h|_\sigma$ are called **Σ-reducts** and written as $\mathcal{A}|_\Sigma$ and $h|_\Sigma$, respectively. In this case, $\_|_\sigma$ coincides with the forgetful functor $U_\Sigma : Alg_{\Sigma'} \to Alg_\Sigma$.

If $\Sigma \subseteq \Sigma'$, a Σ'-algebra $\mathcal{A}'$ is an **extension** of a Σ-algebra $\mathcal{A}$ if $\mathcal{A}'|_\Sigma = \mathcal{A}$.

9.2 Algebras as functors and the Yoneda Lemma

The types over S form the objects of the **free** or **syntactic category over** Σ, $\mathcal{K}(\Sigma)$. The morphisms of $\mathcal{K}(\Sigma)$ are the elements of the quotient $Arr_\Sigma/{\sim_\Sigma}$ where the **Σ-arrow congruence** $\sim_\Sigma$ is the least equivalence relation on Arr_Σ such that the following conditions hold true: Let $I \subseteq \mathcal{I}$.

- For all $f : e_1 \to e_2, g : e_2 \to e_3, h : e_3 \to e_4 \in Arr_\Sigma$, $h \circ (g \circ f) \sim_\Sigma (h \circ g) \circ f$. (1)
- For all $f : e \to e' \in Arr_\Sigma$, $f \circ id_e \sim_\Sigma f$ and $id_{e'} \circ f \sim_\Sigma f$. (2)

- For all $(f_i : e_i \to e)_{i\in I} \in Arr^I_\Sigma$ and $f : \coprod_{i\in I} e_i \to e \in Arr_\Sigma$,

$$[f_i]_{i\in I} \circ \iota_i \sim_\Sigma f_i \quad \text{and} \quad [f \circ \iota_i]_{i\in I} \sim_\Sigma f. \tag{3}$$

- For all $(f_i : e \to e_i)_{i\in I} \in Arr^I_\Sigma$ and $f : e \to \prod_{i\in I} e_i \in Arr_\Sigma$,

$$\pi_i \circ \langle f_i\rangle_{i\in I} \sim_\Sigma f_i \quad \text{and} \quad \langle \pi_i \circ f\rangle_{i\in I} \sim_\Sigma f. \tag{4}$$

- For all $f_1, g_1 : e \to e' \in Arr_\Sigma$ and $f_2, g_2 : e' \to e'' \in Arr_\Sigma$,

$$f_1 \sim_\Sigma g_1 \quad \text{and} \quad f_2 \sim_\Sigma g_2 \quad \text{imply} \quad f_2 \circ f_1 \sim_\Sigma g_2 \circ g_1.$$

- For all $(f_i : e_i \to e)_{i\in I}, (g_i : e_i \to e)_{i\in I} \in Arr^I_\Sigma$,

$$\forall\, i \in I : f_i \sim_\Sigma g_i \quad \text{implies} \quad [f_i]_{i\in I} \sim_\Sigma [g_i]_{i\in I}.$$

- For all $(f_i : e \to e_i)_{i\in I}, (g_i : e \to e_i)_{i\in I} \in Arr^I_\Sigma$,

$$\forall\, i \in I : f_i \sim_\Sigma g_i \quad \text{implies} \quad \langle f_i\rangle_{i\in I} \sim_\Sigma \langle g_i\rangle_{i\in I}.$$

Indeed, 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.

A Σ-algebra $\mathcal{A}$ with carrier A can be regarded as a functor

$$\mathcal{A} : \mathcal{K}(\Sigma) \to Set,$$

which maps each type $e \in \mathcal{T}(S)$ to A_e and each equivalence class $[f : e \to e']_{\sim_\Sigma}$ to $f^{\mathcal{A}} : A_e \to A_{e'}$. The definition of $\sim_\Sigma$ ensures that the mapping is well-defined, i.e., $f \sim_\Sigma g$ implies $f^{\mathcal{A}} = g^{\mathcal{A}}$.

If Σ-algebras are regarded as functors, Lemma 9.2 tells us that Σ-homomorphisms are natural transformations.

The notion of a free or syntactic category and their interpretation by set functors for interpreting signatures has several sources (see, e.g., [148]; [51], Def. 3.7).

Applied to $\mathcal{A} : \mathcal{K}(\Sigma) \to Set$, Lemma 5.1 (see chapter 5) tells us that for $\mathcal{R}(e) = \mathcal{K}(\Sigma)(e, \_)$, $\mathcal{R}'(e) = \mathcal{K}(\Sigma)(\_, e)$ and all $e \in \mathcal{T}_{fo}(S)$,

$$Alg_\Sigma(\mathcal{R}(e), \mathcal{A}) \cong \mathcal{A}(e) \cong Alg_\Sigma(\mathcal{R}'(e), \mathcal{A}), \tag{3}$$

i.e., $\mathcal{A}(e)$ is isomorphic to the set of Σ-homomorphisms from $\mathcal{R}(e)$ (or $\mathcal{R}'(e)$) to $\mathcal{A}$.

As Σ-algebras, $\mathcal{R}(e)$ and $\mathcal{R}'(e)$ are defined as follows:

- For all $e' \in \mathcal{T}_{fo}(S)$, $\mathcal{R}(e)(e') = \mathcal{K}(\Sigma)(e, e')$ and $\mathcal{R}'(e, e') = \mathcal{K}(\Sigma)(e', e)$.
- For all $f : e' \to e'', g : e \to e', h : e'' \to e \in Arr_\Sigma$,

$$f^{\mathcal{R}(e)}([g]_{\sim_\Sigma}) = [f \circ g]_{\sim_\Sigma} \quad \text{and} \quad f^{\mathcal{R}'(e)}([h]_{\sim_\Sigma}) = [h \circ f]_{\sim_\Sigma}.$$

Hence for all $f : e \to e' \in Arr_\Sigma$, a Σ-homomorphism $h : \mathcal{R}(e) \to \mathcal{A}$ maps $[f]_{\sim_\Sigma}$ to an element of $\mathcal{A}(e')$.

By (5), h uniquely represents an element of $\mathcal{A}(e)$ and we have isos

$$\Phi : Alg_\Sigma(\mathcal{R}(e), \mathcal{A}) \to \mathcal{A}(e) \quad \text{and} \quad \Psi : \mathcal{A}(e) \to Alg_\Sigma(\mathcal{R}(e), \mathcal{A})$$

that are defined as follows (see Lemma 5.1):

For all Σ-homomorphisms $h : \mathcal{R}(e) \to \mathcal{A}$, $a \in \mathcal{A}(e)$ and $f : e \to e' \in Arr_\Sigma$,

$$\Phi(h) = h_e([id_e]_{\sim_\Sigma}) \quad \text{and} \quad \Psi(a)_{e'}([f]_{\sim_\Sigma}) = f^{\mathcal{A}}(a). \tag{5}$$

Analogously, for all $f : e' \to e \in Arr_\Sigma$, a Σ-homomorphism $h : \mathcal{R}'(e) \to \mathcal{A}$ maps $[f]_{\sim_\Sigma}$ to an element of $\mathcal{A}(e')$. Again, (5) implies that h uniquely represents an element of $\mathcal{A}(e)$.

Moreover, Corollary 5.2 implies that two types e, e' are $\mathcal{K}(\Sigma)$-isomorphic iff the sets of $\sim_\Sigma$-equivalence classes of Σ-arrows with source (target) e or e', respectively, are naturally equivalent, roughly said: iff e and e' admit the same functions to or from their "environment".

9.3 Σ-terms and -coterms

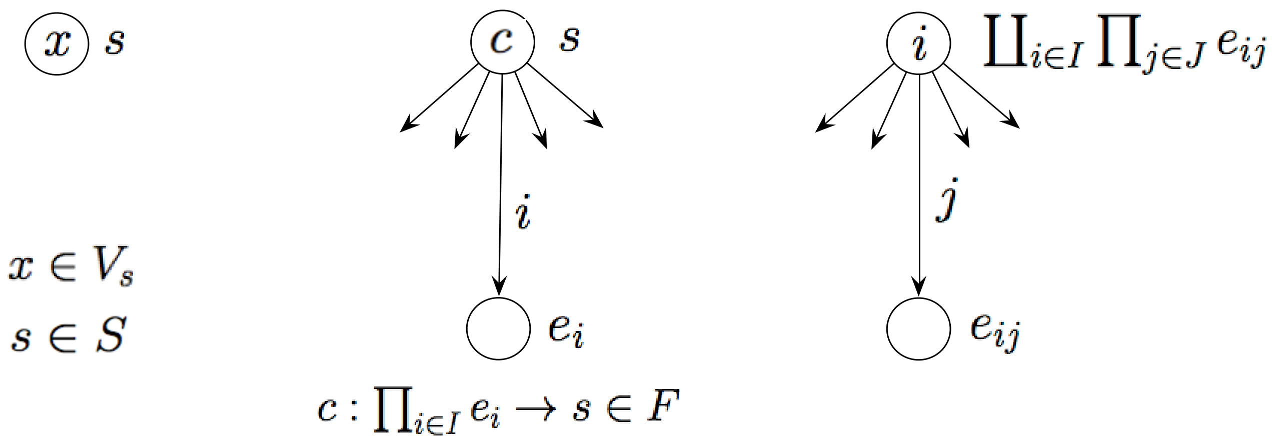
Let $\Sigma = (S, F)$ be a constructive and polynomial signature and V be an S-sorted set of "variables".

The set $CT_\Sigma(V)$ of (**first-order**) **Σ-terms over** V is the greatest $\mathcal{T}_s(S)$-sorted set M of labelled trees over $(\mathcal{I}, \mathcal{I} \cup F \cup V)$ such that for all $I \subseteq \mathcal{I}$, $M_I = I$, and the following conditions hold true:

- For all $s \in S$ and $t \in M_s$, $t \in V_s$ or there are $c : \prod_{i \in I} e_i \to s \in \mathcal{I} \cup F$ and $u \in \times_{i \in I} M_{e_i}$ such that $t = c(u)$, (1)
- for all $e = \coprod_{i \in I} \prod_{j \in J} e_{ij} \in \mathcal{T}_s(S)$ and $t \in M_e$ there are $i \in I$ and $u \in \times_{j \in J} M_{e_{ij}}$ such that $t = i(u)$. (2)

The subset $T_\Sigma(V)$ of $CT_\Sigma(V)$ of **well-founded Σ-terms over** V is the least $\mathcal{T}_s(S)$-sorted set M of well-founded labelled trees over $(\mathcal{I}, \mathcal{I} \cup F \cup V)$ such that for all $I \subseteq \mathcal{I}$, $M_I = I$, and the following conditions hold true:

- For all $s \in S$, $V_s \subseteq M_s$, (3)
- for all $c : \prod_{i \in I} e_i \to s \in \mathcal{I} \cup F$ and $t \in \times_{i \in I} M_{e_i}$, $c(t) \in M_s$, (4)
- for all $e = \coprod_{i \in I} \prod_{j \in J} e_{ij} \in \mathcal{T}_s(S)$, $i \in I$ and $t \in \times_{j \in J} M_{e_{ij}}$, $i(t) \in M_e$. (5)



(1/3) (1/4) (2/5)

Intuitively, Σ-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.

For all $s \in S$, let $V_s = \emptyset$. Then the elements of $CT_\Sigma =_{def} CT_\Sigma(V)$ und $T_\Sigma =_{def} T_\Sigma(V)$ are called **ground Σ-terms**.

If for all $c : e \to s \in F$, e does not contain some index set, then for all $s \in S$, $T_{\Sigma,s}$ is empty.

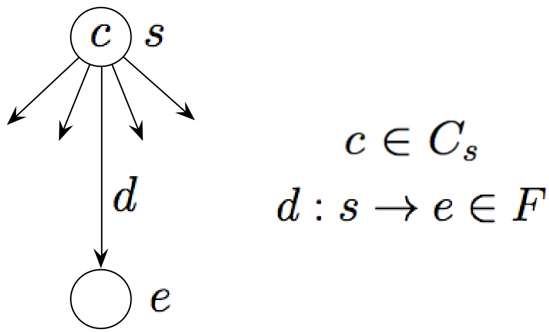
Let $\Sigma = (S, F)$ be a destructive and polynomial signature and C be an S-sorted set of "colors".

The set $DT_\Sigma(C)$ of **Σ-coterms over** C is the greatest $\mathcal{T}_s(S)$-sorted set M of labelled trees over $(\mathcal{I} \cup F, \mathcal{I} \cup C)$ such that for all $I \subseteq \mathcal{I}$, $M_I = I$, (2) holds true and

- for all $s \in S$ and $t \in M_s$ there are $c \in C_s$ and $u \in \bigtimes_{d:s\to e\in F} M_e$ such that $t = c(u)$. (6)

The subset $coT_\Sigma(C)$ of $DT_\Sigma(C)$ of **well-founded Σ-coterms over** C is the least $\mathcal{T}_s(S)$-sorted set M of well-founded labelled trees over $(\mathcal{I} \cup F, \mathcal{I} \cup C)$ such that for all $I \subseteq \mathcal{I}$, $M_I = I$, (5) holds true and

- for all $s \in S$, $c \in C_s$ and $u \in \bigtimes_{d:s\to e\in F} M_e$, $c(u) \in M_s$. (7)



(6/7)

Intuitively, Σ-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.

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 Σ-coterms**.

If for all $d \in F$, $trg(d)$ does not contain some index set, then for all $s \in S$, $DT_{\Sigma,s}$ is a singleton.

9.4 Sample terms and coterms

Let $\Sigma = Nat$.

$CT_{\Sigma,nat}$ is the greatest subset M of $ltr(1, \{zero : 1 \rightarrow nat, succ : nat \rightarrow nat\})$ with the following property:

- For all $t \in M$, $t = zero$ or $t = succ\{() \rightarrow u\}$ for some $u \in M$.

T_Σ is the least subset M of $ltr(1, \{zero : 1 \to nat, succ : nat \to nat\})$ with the following properties:

- $zero \in M$.
- For all $t \in M$, $succ\{() \to t\} \in M$.

Hence $T_\Sigma \cong \mathbb{N}$ and $CT_\Sigma \cong \mathbb{N} \cup \{\omega\}$.

Let $\Sigma = coNat$.

$DT_{\Sigma,nat}$ is the greatest subset M of $ltr(\{(), pred : nat \to nat + 1\}, \{(), 1, 2\})$ with the following property:

- For all $t \in M$, $t = ()\{pred \to 2\}$ or $t = ()\{pred \to 1\{() \to u\}\}$ for some $u \in M$.

Hence $DT_\Sigma \cong \mathbb{N} \cup \{\omega\}$.

Let $\Sigma = List(X)$.

$CT_{\Sigma,state}$ is the greatest subset M of

$$ltr(\{1,2\},\{\alpha:1\to state, cons:X\times state\to state\}\cup X)$$

with the following property:

- For all $t\in M$, $t=\alpha$ or $t=cons\{1\to x,2\to u\}$ for some $x\in X$ and $u\in M$.

T_Σ is the least subset M of

$$ltr(\{1,2\},\{\alpha:1\to state, cons:X\times state\to state\}\cup X)$$

with the following properties:

- $\alpha\in M$.
- For all $x\in X$ and $t\in M$, $cons\{1\to x,2\to t\}\in M$.

Hence $T_\Sigma\cong X^*$ and $CT_\Sigma\cong X^\infty=X^*\cup X^{\mathbb{N}}$.

Let $\Sigma=coList(X)$.

$DT_{\Sigma,state}$ is the greatest subset M of $ltr(\{split:state\to X\times state+1,1,2\},1\cup X)$ with the following property:

- For all $t \in M$, $t = ()\{split \to 2\}$ or $t = ()\{split \to 1\{1 \to x, 2 \to u\}\}$ for some $x \in X$ and $u \in M$.

Hence $DT_{\Sigma,state} \cong X^\infty = X^* \cup X^{\mathbb{N}}$.

Let $\Sigma = Reg(X)$.

$T_{\Sigma,state}$ is the least subset M of $ltr(\{1,2\}, \{par, seq, star, \overline{\_}, \widehat{\_}\} \cup \mathcal{P}_+(X))$ with the following properties:

- For all $t, u \in M$, $B \in \mathcal{P}_+(X)$ and $c \in 2$,

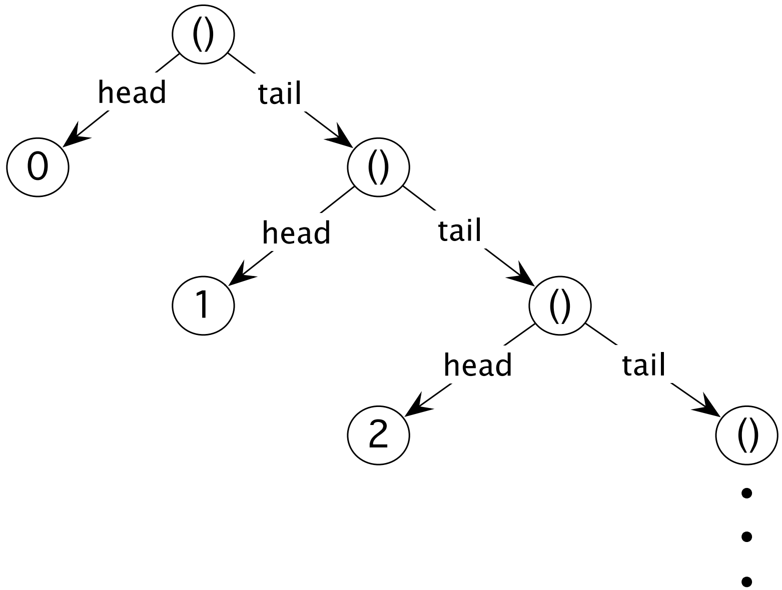
$$par(t,u),\ seq(t,u),\ star(t),\ \overline{B},\ \widehat{c} \in M.$$

Let $\Sigma = Stream(X)$.

$DT_{\Sigma,state}$ is the greatest subset M of $ltr(\{head, tail\}, 1 \cup X)$ with the following property:

- For all $t \in M$, $t = ()\{head \to x, tail \to u\} \in M$ for some $x \in X$ and $u \in M$.

Hence $DT_{\Sigma,state} \cong X^{\mathbb{N}}$.



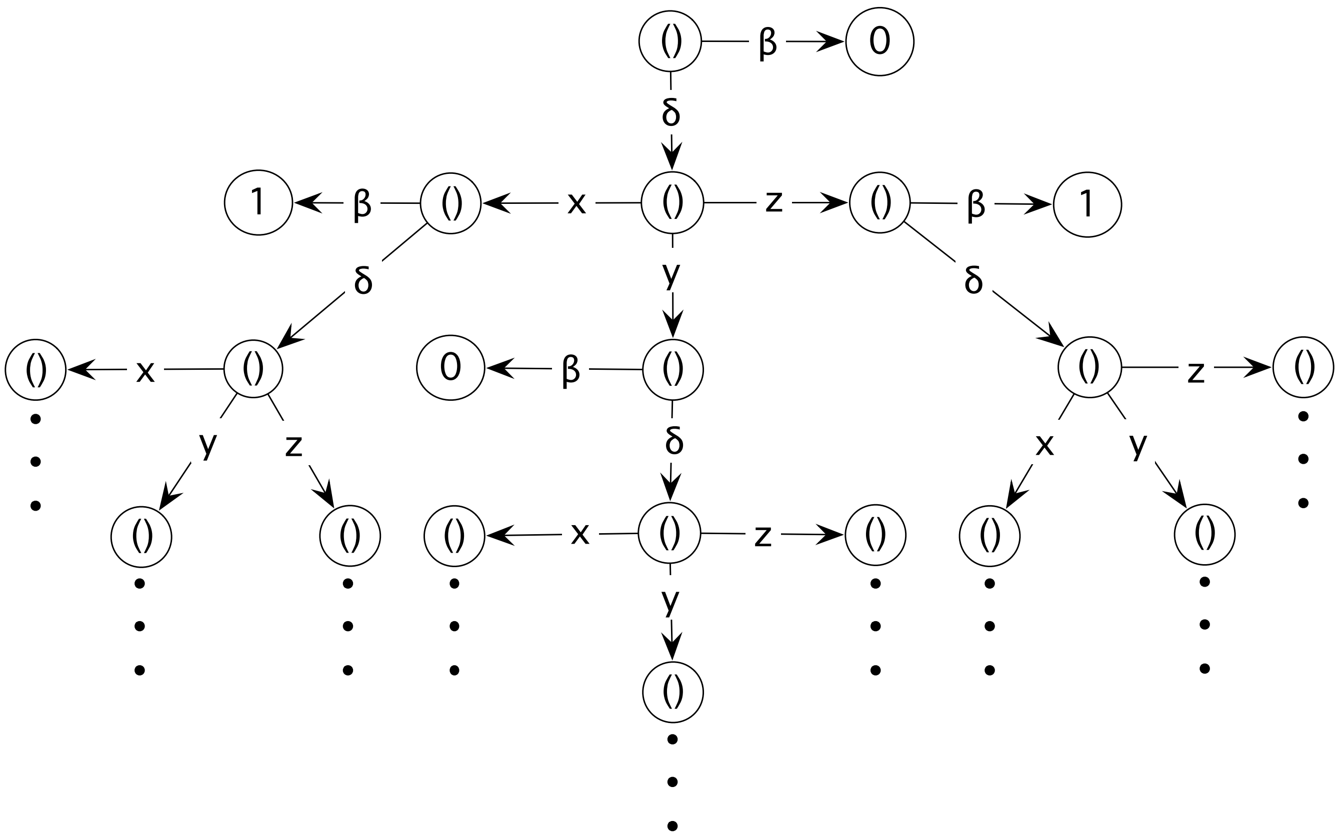
Stream$(\mathbb{N})$-*coterm that represents the stream of natural numbers.*

Let $\Sigma = DAut(X,Y)$.

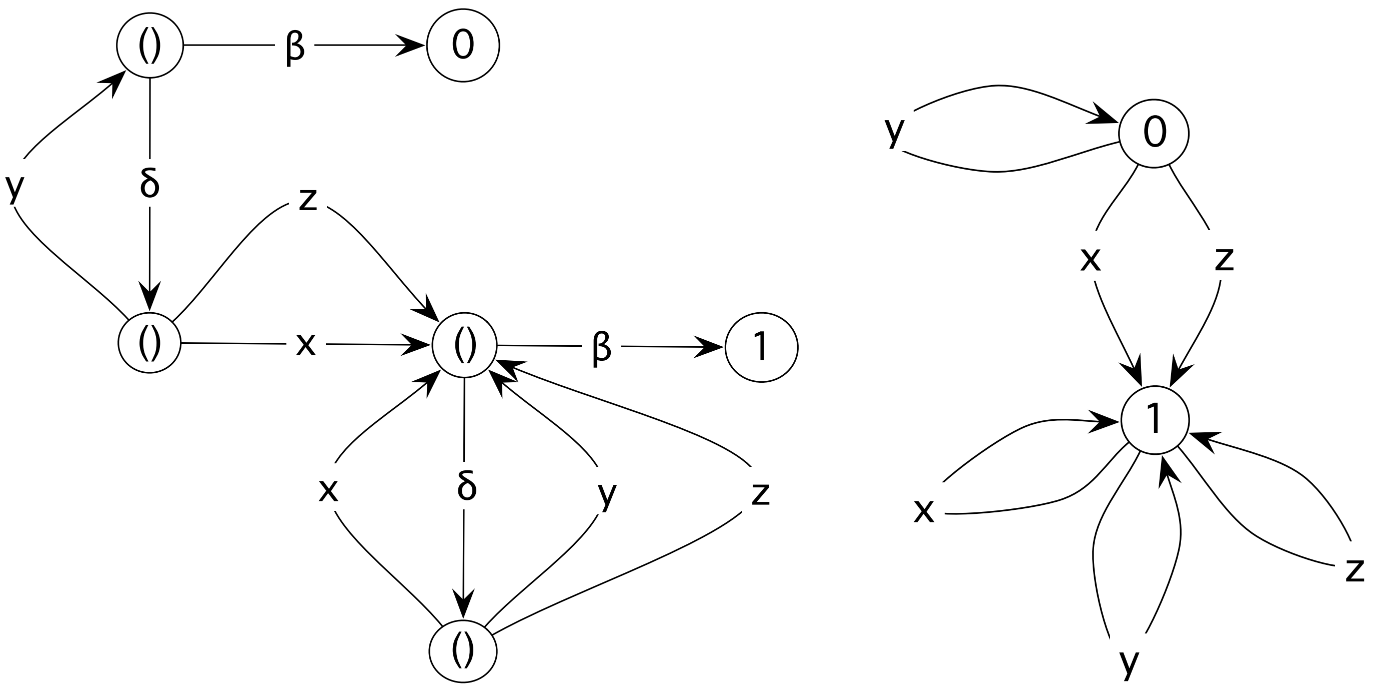
$DT_{\Sigma,state}$ is the greatest subset M of $ltr(\{\delta,\beta\}\cup X, 1\cup Y)$ with the following property:

- For all $t\in M$, $t = ()\{\delta\to ()\{x\to t_x \mid x\in X\}, \beta\to y\}$ for some $(t_x)_{x\in X}\in M^X$ and $y\in Y$.

Hence $DT_{\Sigma,state}\cong Y^{X^*}$.



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 coterm into a finite graph (left) and the corresponding transition graph (right)

9.5 Term and coterm algebras

Let $\Sigma = (S, C)$ be a constructive polynomial signature and $V \in Set^S$.

$CT_\Sigma(V)$ is a Σ-algebra:

- For all $I \subseteq \mathcal{I}$, $CT(V)_I = I$.
- For all $c : \prod_{i \in I} e_i \to s \in C$ and $t \in \prod_{i \in I} e_i$, $c^{CT_\Sigma(V)}(t) =_{def} c(t)$.
- For all $e = \coprod_{i \in I} e_i \in \mathcal{T}_s(S)$, $CT_\Sigma(V)_e =_{def} \{i(t) \mid i \in I,\ t \in CT_\Sigma(V)_{e_i}\}$,
- For all $e = \prod_{i \in I} e_i \in \mathcal{T}_p(S)$, $CT_\Sigma(V)_e =_{def} \times_{i \in I} CT_\Sigma(V)_{e_i}$.

Hence the carrier of $CT_\Sigma(V)$ is a model of $\mathcal{T}_{po}(S)$.

Moreover, $T_\Sigma(V)$ is a Σ-subalgebra of $CT_\Sigma(V)$.

Let $\Sigma = (S, D)$ be a destructive polynomial signature and $V \in Set^S$.

$DT_\Sigma(C)$ is a Σ-algebra:

- For all $I \subseteq \mathcal{I}$, $DT(C)_I = I$.
- For all $d : s \to e \in D$ and $t \in DT_\Sigma(C)_s$, $d^{DT_\Sigma(C)}(t) =_{def} \lambda w.t(dw)$.
- For all $e = \coprod_{i \in I} e_i \in \mathcal{T}_s(S)$, $DT_\Sigma(C)_e =_{def} \{i(t) \mid i \in I,\ t \in DT_\Sigma(C)_{e_i}\}$,

- For all $e = \prod_{i\in I} e_i \in \mathcal{T}_p(S)$, $DT_\Sigma(C)_e =_{def} \times_{i\in I} DT_\Sigma(C)_{e_i}$.

Hence the carrier of $DT_\Sigma(C)$ is a model of $\mathcal{T}_{po}(S)$.

Moreover, $coT_\Sigma(C)$ is a Σ-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 $d : s \to e$ the **derivatives** $d^{DT_\Sigma(C)}(t)$ and $d^{DT_\Sigma(C)}(t')$ coincide (see sample algebra 9.6.23 and section 14.1.)

9.6 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 *Nat*-algebra $\mathbb{N}$ whose operations

$$zero^{\mathbb{N}} : 1 \to \mathbb{N}, \; succ^{\mathbb{N}} : \mathbb{N} \to \mathbb{N}$$

are defined as follows: For all $n \in \mathbb{N}$,

$$zero^{\mathbb{N}}() = 0,$$
$$succ^{\mathbb{N}}(n) = n+1.$$

$\mathbb{N}$ is also the carrier of the $List(X)$-algebra *Length* whose operations

$$\alpha^{Length} : 1 \to \mathbb{N},\ cons^{Length} : X \times \mathbb{N} \to \mathbb{N}$$

are defined as follows: For all $x \in X$ and $n \in \mathbb{N}$,

$$\alpha^{Length}(x) = 0,$$
$$cons^{Length}(x,n) = n+1.$$

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

$$pred^{\mathbb{N}_\infty} : \mathbb{N}_\infty \to \mathbb{N}_\infty + 1 \quad \text{resp.} \quad pred^{1^\infty} : 1^\infty \to 1^\infty + 1$$

are defined as follows: For all $n > 0$,

$$\begin{array}{lll} pred^{\mathbb{N}_\infty}(0) & = & (), \\ pred^{\mathbb{N}_\infty}(n) & = & n-1, \\ pred^{\mathbb{N}_\infty}(\infty) & = & \infty, \end{array} \qquad \begin{array}{lll} pred^{1^\infty}(\epsilon) & = & (), \\ pred^{1^\infty}(()^n) & = & ()^{n-1}, \\ pred^{1^\infty}(\lambda n.()) & = & \lambda n.(). \end{array}$$

3. The set $X^* \times Y$ of **guarded sequences** is the carrier of the $Dyn(X,Y)$-algebra $Seq(X,Y)$ whose operations

$$cons^{Seq(X,Y)} : X \times (X^* \times Y) \to X^* \times Y,\ \alpha^{Seq(X,Y)} : Y \to X^* \times Y$$

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 [69] 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

$$cons^{X^*} : X \times X^* \to X^*, \ \alpha^{X^*} : 1 \to X^*$$

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

$$one^{Word(X)} : 1 \to X^*, \ mul^{Word(X)} : X^* \times X^* \to X^*$$

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 *Mon*-algebra *Endo(X)* whose operations

$$one^{Endo(X)} : 1 \to X^X, \;\; mul^{Endo(X)} : X^X \times X^X \to X^X$$

are defined follows: For all $f, g : X \to X$,

$$\begin{array}{rcl} one^{Endo(X)} & = & id_X, \\ mul^{Endo(X)}(f,g) & = & g \circ f. \end{array}$$

Given a $Med(X)$-algebra $\mathcal{A}$ with carrier A, A^A is the carrier of the $List(X)$-algebra $Reach(\mathcal{A})$ whose operations

$$cons^{Reach(\mathcal{A})} : X \times A^A \to A^A, \;\; \alpha^{Reach(\mathcal{A})} : 1 \to A^A$$

are defined as follows: For all $x \in X$, $f \in A^A$ and $a \in A$,

$$\begin{array}{rcl} cons^{Reach(\mathcal{A})}(x,f) & = & f \circ \lambda a.\delta^{\mathcal{A}}(a)(x), \\ \alpha^{Reach(\mathcal{A})}() & = & id_A. \end{array}$$

Exercise 12 The function

$$\begin{array}{rcl} reach_{state} : X^* & \to & Reach(\mathcal{A}) \\ \epsilon & \mapsto & id_A \\ xw & \mapsto & reach_{state}(w) \circ \lambda a.\delta^{\mathcal{A}}(a)(x) \qquad x \in X, \;\; w \in X^* \end{array}$$

is $List(X)$-homomorphic. ❑

Consequently, Theorem 9.7 implies $reach_{state} = fold^{\mathcal{A}}$ because X^* is initial in $Alg_{List(X)}$ (see sample initial algebra 9.13.3).

5. $X^{\mathbb{N}}$ is the carrier of the *Stream*(X)-algebra *InfSeq*(X) whose operations

$$head^{InfSeq(X)} : X^{\mathbb{N}} \to X, \; tail^{InfSeq(X)} : X^{\mathbb{N}} \to X^{\mathbb{N}}$$

are defined as follows: For all $f : \mathbb{N} \to X$,

$$\begin{array}{rcl} head^{InfSeq(X)}(f) & = & f(0), \\ tail^{InfSeq(X)}(f) & = & \lambda n.f(n+1). \end{array}$$

$X^{\mathbb{N}}$ is also the carrier of the *coStream(X)*-algebra *coInfSeq*(X) whose operation

$$\delta^{coInfSeq(X)} : X \times X^{\mathbb{N}} \to X^{\mathbb{N}}$$

is defined as follows: For all $x \in X$, $f : \mathbb{N} \to X$ and $n > 0$,

$$\begin{array}{rcl} \delta^{coInfSeq(X)}(x, f)(0) & = & x, \\ \delta^{coInfSeq(X)}(x, f)(n) & = & f(n-1). \end{array}$$

6. The following $Stream(\mathbb{Z})$-algebra zo represents the periodic streams $0, 1, 0, 1, \ldots$ and $1, 0, 1, 0, \ldots$:

$$
\begin{aligned}
zo_{list} &= \{blink, blink'\}, \\
head^{zo}(blink) &= 0, \\
tail^{zo}(blink) &= blink', \\
head^{zo}(blink') &= 1, \\
tail^{zo}(blink') &= blink.
\end{aligned}
$$

7. The following $DAut(\mathbb{Z}, 2)$-algebra eo is the minimal automaton that accepts a word $x_1 \ldots x_n$ over $\mathbb{Z}$ in $esum$ or $osum$ iff $\sum_{i=1}^{n} x_i$ is even or odd, respectively (see example 9.17):

$$
\begin{aligned}
eo_{state} &= \{esum, osum\}, \\
\delta^{eo}(esum) &= \lambda x.if\ even(x)\ then\ esum\ else\ osum, \\
\delta^{eo}(osum) &= \lambda x.if\ odd(x)\ then\ esum\ else\ osum, \\
\beta^{eo} &= \lambda st.if\ st = esum\ then\ 1\ else\ 0.
\end{aligned}
$$

8. $T = X^* \times Y \cup X^{\mathbb{N}}$ is the carrier of the *coDyn*(X,Y)-algebra *coSeq*(X,Y) whose operation

$$split^{coSeq(X,Y)} : T \to X \times T + Y$$

is defined as follows: For all $x \in X$, $w \in X^*$, $y \in Y$ and $f \in X^{\mathbb{N}}$,

$$\begin{array}{rcl} split^{coSeq(X,Y)}(\epsilon, y) & = & \iota_2(y), \\ split^{coSeq(X,Y)}(xw, y) & = & \iota_1(x, (w, y)), \\ split^{coSeq(X,Y)}(f) & = & \iota_1(f(0), \lambda n.f(n+1)). \end{array}$$

See [69] 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

$$split^{Neseq(X)} : T \to X \times T + 1$$

is defined as follows: For all $x \in X$, $w \in X^+$ and $f \in X^{\mathbb{N}}$,

$$\begin{array}{rcl} split^{Neseq(X)}(x) & = & \iota_2(), \\ split^{Neseq(X)}(xw) & = & \iota_1(x, w), \\ split^{Neseq(X)}(f) & = & \iota_1(f(0), \lambda n.f(n+1)). \end{array}$$

10. $T = ftr(2, X)$ (see chapter 2) is the carrier of the $Bintree(X)$-algebra $FBin(X)$ whose operations

$$bjoin^{FBin(X)} : X \times T \times T \to T, \qquad empty^{FBin(X)} : 1 \to T$$

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 $infBintree(X)$-algebra $InfBin(X)$ whose operations

$$left, right : T \to T, \qquad root^{InfBin(X)} : T \to X$$

are defined as follows: For all $t \in T$,

$$\begin{aligned} left^{InfBin(X)}(t) &= \lambda w.t(0w), \\ right^{InfBin(X)}(t) &= \lambda w.t(1w), \\ root^{InfBin(X)}(t) &= t(\epsilon). \end{aligned}$$

12. $T = ltr(2, X)$ (see chapter 2) is the carrier of the $coBintree(X)$-algebra $Bin(X)$ whose operation

$$split^{Bin(X)} : T \to T \times X \times T + 1$$

is defined as follows: For all $x \in X$ and $t, u \in T$,

$$\begin{array}{rcl} split^{Bin(X)}(x\{0 \to t, 1 \to u\}) & = & \iota_1(x, t, u), \\ split^{Bin(X)}(\Omega) & = & \iota_2(). \end{array}$$

13. Let $T = otr(\mathbb{N}, X) \cap ftr(\mathbb{N}, X)$. (T, T^*) is the carrier of the $Tree(X)$-algebra $FTree(X)$ whose operations

$$join^{FTree(X)} : X \times T^* \to T, \ cons^{FTree(X)} : T \times T^* \to T^*, \ nil^{FTree(X)} : 1 \to T^*$$

are defined as follows: For all $x \in X$, $n > 0$, $t, t_1, \dots, t_n \in T$,

$$\begin{array}{rcl} join^{FTree(X)}(x, \epsilon) & = & x, \\ join^{FTree(X)}(x, (t_1, \dots, t_n)) & = & x(t_1, \dots, t_n), \\ nil^{FTree(X)} & = & \epsilon, \\ cons^{FTree(X)}(t, \epsilon) & = & t, \\ cons^{FTree(X)}(t, (t_1, \dots, t_n)) & = & (t, t_1, \dots, t_n). \end{array}$$

14. $T = otr(\mathbb{N}, X) \cap wtr(\mathbb{N}, X)$ is the carrier of the $Tree_\omega(X)$-algebra $WFTree(X)$ whose operation

$$join^{WFTree(X)} \;:\; X \times T^\infty \to T$$

is defined as follows:

For all $x \in X$, $n > 0$, $t_1, \dots, t_n \in T$ and $f \in T^{\mathbb{N}}$,

$$\begin{array}{rcl} join^{WFTree(X)}(x, \epsilon) &=& x, \\ join^{WFTree(X)}(x, (t_1, \dots, t_n)) &=& x(t_1, \dots, t_n), \\ join^{WFTree(X)}(x, f) &=& x\{n \to f(n) \mid n \in \mathbb{N}\}. \end{array}$$

15. $T = ftr(Y, X)$ is the carrier of the $ETree(X)$-algebra $FETree(X)$ whose operation

$$join^{FETree(X)} \;:\; X \times (Y \times T)^* \to T$$

is defined as follows: For all $x \in X$, $n > 0$ and $y_1, \dots, y_n \in Y$ and $t_1, \dots, t_n \in T$,

$$\begin{array}{rcl} join^{FETree(X)}(x, \epsilon) &=& x, \\ join^{FETree(X)}(x, ((y_1, t_1), \dots, (y_n, t_n)) &=& x\{y_1 \to t_1, \dots, y_n \to t_n\}. \end{array}$$

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

$$\begin{aligned} subtrees^{FBInfTree(X)} &: T \to T^+, \\ root^{FBInfTree(X)} &: T \to X \end{aligned}$$

are defined as follows: For all $x \in X$, $n > 0$ and $t_1, \ldots, t_n \in T$,

$$\begin{aligned} subtrees^{FBInfTree(X)}(x(t_1, \ldots, t_n)) &= (t_1, \ldots, t_n), \\ root^{FBInfTree(X)}(x(t_1, \ldots, t_n)) &= x. \end{aligned}$$

17. $T = otr(\mathbb{N}, X) \cap fbtr(\mathbb{N}, X)$ is the carrier of the $coTree_\omega(X)$-algebra $FBTree(X)$ whose operations

$$\begin{aligned} subtrees^{FBTree(X)} &: T \to T^*, \\ root^{FBTree(X)} &: T \to X \end{aligned}$$

are defined as follows: For all $x \in X$, $n > 0$ and $t_1, \ldots, t_n \in T$,

$$\begin{aligned} subtrees^{FBTree(X)}(x) &= \epsilon, \\ subtrees^{FBTree(X)}(x(t_1, \ldots, t_n)) &= (t_1, \ldots, t_n), \\ root^{FBTree(X)}(x) &= x, \\ root^{FBTree(X)}(x(t_1, \ldots, t_n)) &= x. \end{aligned}$$

18. Let $T = otr(\mathbb{N}, X)$. The $coTree(X)$-algebra $Tree_\infty(X)$ is defined as follows:

$$\begin{array}{rcl} Tree_\infty(X)_{tree} & = & T, \\ Tree_\infty(X)_{trees} & = & T^\infty, \\ subtrees^{Tree_\infty(X)} & : & T \to T^\infty, \\ root^{Tree_\infty(X)} & : & T \to X, \\ split^{Tree_\infty(X)} & : & T^\infty \to T \times T^\infty + 1 \end{array}$$

are defined as follows:

For all $x \in X$, $t = x\{i \to t_i \mid i \in def(t) \cap \mathbb{N}\}$, $w \in T^*$ and $f \in T^{\mathbb{N}}$,

$$\begin{array}{rcl} subtrees^{Tree_\infty(X)}(t) & = & (t_i)_{i \in def(t) \cap \mathbb{N}}, \\ root^{Tree_\infty(X)}(t) & = & x, \\ split^{Tree_\infty(X)}(t \cdot w) & = & \iota_1(t, w), \\ split^{Tree_\infty(X)}(f) & = & \iota_1(f(0), \lambda n.f(n+1)), \\ split^{Tree_\infty(X)}(\epsilon) & = & \iota_2(). \end{array}$$

19. Let $\Sigma = Reg(X)$. The set $\mathcal{P}(X^*)$ of **languages** is the carrier of the $Reg(X)$-algebra $Lang(X)$ whose operations

$$\begin{array}{rcl} par^{Lang(X)}, seq^{Lang(X)} & : & \mathcal{P}(X^*) \times \mathcal{P}(X^*) \to \mathcal{P}(X^*), \\ star^{Lang(X)} & : & \mathcal{P}(X^*) \to \mathcal{P}(X^*), \\ \overline{\_}^{Lang(X)} & : & \mathcal{P}_+(X) \to \mathcal{P}(X^*), \\ \widehat{\_}^{Lang(X)} & : & 2 \to \mathcal{P}(X^*). \end{array}$$

are defined as follows:

For all $L, L' \subseteq X^*$ and $B \in \mathcal{P}_+(X)$,

$$\begin{array}{rcl} par^{Lang(X)}(L, L') & = & L \cup L', \\ seq^{Lang(X)}(L, L') & = & L \cdot L', \\ star^{Lang(X)}(L) & = & L^*, \\ \overline{B}^{Lang(X)} & = & B, \\ \widehat{0}^{Lang(X)} & = & \emptyset, \\ \widehat{1}^{Lang(X)} & = & \{\epsilon\}. \end{array}$$

The usual semantics of a regular expression, i.e., a ground $Reg(X)$-term, say t, is obtained by *folding* t in $Lang(X)$ (see sample initial algebra 9.13.6).

20. $\mathcal{P}(X^*)$ is also the carrier of the $Acc(X)$-algebra $Pow(X)$ whose operations

$$\begin{aligned} \delta^{Pow(X)} &: \mathcal{P}(X^*) \to \mathcal{P}(X^*)^X, \\ \beta^{Pow(X)} &: \mathcal{P}(X^*) \to 2 \end{aligned}$$

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

$$\begin{aligned} \delta^{NPow(X)} &: \mathcal{P}(X^*) \to \mathcal{P}_\omega(\mathcal{P}(X^*))^X, \\ \beta^{NPow(X)} &: \mathcal{P}(X^*) \to 2 \end{aligned}$$

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

$$\begin{array}{rcl} par^{Bool}, seq^{Lang(X)} & : & 2 \times 2 \to 2, \\ star^{Bool} & : & 2 \to 2, \\ \underline{\ \ }^{Bool} & : & \mathcal{P}(X) \to 2, \\ \widehat{\underline{\ \ }}^{Bool} & : & 2 \to 2 \end{array}$$

are defined as follows: For all $x, y \in 2$ and $B \in \mathcal{P}_+(X)$,

$$\begin{array}{rcl} par^{Bool}(x,y) & = & max\{x,y\}, \\ seq^{Bool}(x,y) & = & x * y, \\ star^{Bool}(x) & = & 1, \\ \underline{\ \ }^{Bool}(B) & = & 0, \\ \widehat{x}^{Bool} & = & x. \end{array}$$

23. From now on, we often write $t_1 + \cdots + t_n$ and $t_1 * \cdots * t_n$ for $Reg(X)$-terms ("regular expressions"; see section 9.4) of the form

$$par(\ldots(par(t_1, t_2), \ldots), t_n) \text{ and } seq(\ldots(seq(t_1, t_2), \ldots), t_n),$$

respectively (see section 8.2). Muliplication priorizes over addition.

$T_{Reg(X)}$ is the carrier of both the $Reg(X)$-algebra of ground $Reg(X)$-terms and the **Brzozowski automaton** $Bro(X)$ for accepting regular languages [36, 92], i.e., the $Acc(X)$-algebra whose operations

$$\delta = \delta^{Bro(X)} : T_{Reg(X)} \to T_{Reg(X)}^X \quad \text{and} \quad \beta = \beta^{Bro(X)} : T_{Reg(X)} \to 2$$

are inductively defined as follows: For all $t, u \in T_{Reg(X)}$, $B \in \mathcal{P}_+(X)$ and $c \in 2$,

$$\begin{aligned}
\delta(t+u) &= \lambda x.(\delta(t)(x) + \delta(u)(x)), \\
\delta(t*u) &= \lambda x.(\delta(t)(x) * u + \widehat{\beta(t)} * \delta(u)(x)), \\
\delta(star(t)) &= \lambda x.(\delta(t)(x) * star(t)), \\
\delta(\overline{B}) &= \lambda x.\widehat{x \in B}, \\
\delta(\widehat{c}) &= \lambda x.\widehat{0}, \\
\beta(t+u) &= max(\beta(t), \beta(u)), \\
\beta(t*u) &= \beta(t) * \beta(u), \\
\beta(star(t)) &= 1, \\
\beta(\overline{B}) &= 0, \\
\beta(\widehat{c}) &= c.
\end{aligned}$$

For a proof that these equations define δ and β uniquely on $T_{Reg(X)}$, see sample inductive definition 16.3.20.

$\delta(t)(x)$ and $\beta(t)$ are called the x-*derivative* and *initial value* of t, respectively (see [36, 92]).

24. The set Y^{X^*} of **behavior functions** is the carrier of the $DAut(X,Y)$-algebra $Beh(X,Y)$ whose operations

$$\begin{aligned}\delta^{Beh(X,Y)} &: Y^{X^*} \to (Y^{X^*})^X,\\ \beta^{Beh(X,Y)} &: Y^{X^*} \to Y\end{aligned}$$

are defined as follows: For all $f : X^* \to Y$ and $x \in X$,

$$\begin{aligned}\delta^{Beh(X,Y)}(f)(x) &= \lambda w.f(x \cdot w),\\ \beta^{Beh(X,Y)}(f) &= f(\epsilon).\end{aligned}$$

Exercise 13 Show that the characteristic function $\chi : \mathcal{P}(X^*) \to 2^{X^*}$ (see chapter 2) is an $Acc(X)$-isomorphism from $Pow(X)$ to $Beh(X,2)$. ❏

χ is also a $Reg(X)$-isomorphism from $Lang(X)$ to $RegBeh(X)$ where $RegBeh(X)$ has the carrier 2^{X^*} and interprets the operations of $Reg(X)$ as follows:

For all $f, g : X^* \to 2$, $w \in X^*$, $v \in X^+$, $B \in \mathcal{P}_+(X)$ and $c \in 2$,

$$\begin{array}{rcl}
par^{RegBeh(X)}(f,g)(w) & = & max\{f(w), g(w)\}, \\
seq^{RegBeh(X)}(f,g)(w) & = & max(\left\{\begin{array}{l} \{f(\epsilon) * g(w) \mid w \in X^+\} \cup \\ \{f(w) * g(\epsilon) \mid w \in X^+\} \cup \\ \{f(w_1) * g(w_2) \mid w_1, w_2 \in X^+,\ w_1 w_2 = w\} \end{array}\right\}), \\
star^{RegBeh(X)}(f)(\epsilon) & = & 1, \\
star^{RegBeh(X)}(f)(v) & = & max\{f(w_1) * \cdots * f(w_n) \mid w_1, \ldots, w_n \in X^+,\ w_1 \ldots w_n = v\}, \\
\underline{\ \ }^{RegBeh(X)}(B)(w) & = & if\ \ w \in B\ then\ 1\ else\ 0, \\
\widehat{c}^{RegBeh(X)}(w) & = & if\ \ c = 1 \land w = \epsilon\ then\ 1\ else\ 0.
\end{array}$$

$RegBeh(X)$ is implemented in `Compiler.hs` under the name `regBeh`.

25. A commutative monoid $(M, +, 0)$ is a semiring if there are a constant $1 \in M$ and a function $* : M^2 \to M$ 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 $\cdot : R \times A \to A$ 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 $0 \cdot a = 0 = r \cdot 0$ (multiplication with 0 annihilates A or R).

Every semiring R is an R-semimodule. (1)

Given a set X, R^X and R^X_ω are also R-semimodules where addition, zero and scalar multiplication are defined as follows: For all $f, g \in R^X_R$, $x \in X$ 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 $h : A \to B$ 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 δ,β 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 [155]).

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

$$\begin{aligned}\delta^{MBeh(X,Y)} &: Y^{X^+} \to (Y^{X^+})^X,\\ \beta^{MBeh(X,Y)} &: Y^{X^+} \to Y^X\end{aligned}$$

are defined as follows: For all $f : X^+ \to Y$, $x \in X$ 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 $\mathcal{C}(X,Y)$ 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

$$\begin{aligned}\delta^{Causal(X,Y)} &: \mathcal{C}(X,Y) \to \mathcal{C}(X,Y)^X,\\ \beta^{Causal(X,Y)} &: \mathcal{C}(X,Y) \to Y^X\end{aligned}$$

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

$$\begin{aligned}\delta^{PBeh(X,Y)} &: ltr(X,Y) \to (1 + ltr(X,Y))^X,\\ \beta^{PBeh(X,Y)} &: ltr(X,Y) \to Y\end{aligned}$$

are defined as follows: For all $Z \subseteq X$, $t = y\{x \to t_x \mid x \in Z\} \in ltr(X,Y)$ 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. $T = otr(X \times \mathbb{N}, Y)$ is the carrier of the $NAut^*(X,Y)$-algebra $NBeh(X,Y)$ whose operations

$$\delta^{NBeh(X,Y)} : T \to (T^*)^X, \qquad \beta^{NBeh(X,Y)} : T \to Y$$

are defined as follows:

For all $\{n_x \mid x \in X\} \subseteq \mathbb{N}$, $t = y\{(x,i) \to t_{x,i} \mid x \in X,\ 1 \leq i \leq n_x\} \in T$ and $x \in X$,

$$\begin{array}{rcl} \delta^{NBeh(X,Y)}(t)(x) & = & (t_{x,1}, \ldots, t_{x,n_x}), \\ \beta^{NBeh(X,Y)}(t) & = & y. \end{array}$$

Let $\Sigma = (S, C)$ be a finitary signature. For all $c : s_1 \times \cdots \times s_n \to s \in C$, S-sorted subsets L of $T_{\Sigma,s}$ (see section 9.3) and $1 \leq i \leq n$,

$$sucs(c, L)_i \ =_{def} \ \{t_i \in T_{\Sigma,s_i} \mid \forall\ j \in [n] \setminus \{i\}\ \exists\ t_j \in T_{\Sigma,s_j} : c(t_1, \ldots, t_n) \in L\}.$$

29. $\mathcal{P}(T_\Sigma) =_{def} (\mathcal{P}(T_{\Sigma,s}))_{s \in S \cup \mathcal{P}(\mathcal{I})}$ is the carrier of the $TAcc(\Sigma)$-algebra $TPow(\Sigma)$ whose operations

$$\delta_c^{TPow(\Sigma)} : \mathcal{P}(T_{\Sigma,s}) \to \mathcal{P}(T_{\Sigma,s_1}) \times \ldots \times \mathcal{P}(T_{\Sigma,s_n}), \qquad c : s_1 \times \cdots \times s_n \to s \in C,$$

are defined as follows: For all $L \subseteq T_{\Sigma,s}$, $\delta_c^{TPow(\Sigma)}(L) = (sucs(c, L)_1, \ldots, sucs(c, L)_n)$.

30. $\mathcal{P}(T_\Sigma)$ is also the carrier of the $NTAcc(\Sigma)$-algebra $NTPow(\Sigma)$ whose operations

$$\delta_c^{NTPow(\Sigma)} : \mathcal{P}(T_{\Sigma,s}) \to \mathcal{P}_\omega(\mathcal{P}(T_{\Sigma,s_1}) \times \ldots \times \mathcal{P}(T_{\Sigma,s_n})), \qquad c : s_1 \times \cdots \times s_n \to s \in C,$$

are defined as follows: For all $L \subseteq T_{\Sigma,s}$, $\delta_c^{NTPow(\Sigma)}(L) = \{(sucs(c,L)_1, \ldots, sucs(c,L)_n)\}$.

31. $\mathcal{P}(T_\Sigma)$ is also the carrier of the $NTAcc^*(\Sigma)$-algebra $NTPow^*(\Sigma)$ whose operations

$$\delta_c^{NTPow(\Sigma)} : \mathcal{P}(T_{\Sigma,s}) \to (\mathcal{P}(T_{\Sigma,s_1}) \times \ldots \times \mathcal{P}(T_{\Sigma,s_n}))^*, \qquad c : s_1 \times \cdots \times s_n \to s \in C,$$

are defined as follows: For all $L \subseteq T_{\Sigma,s}$, $\delta_c^{NTPow^*(\Sigma)}(L) = [(sucs(c,L)_1, \ldots, sucs(c,L)_n)]$.

32. $T = otr(X \times \mathbb{N}, 1)$ is the carrier of the $NMed^*(X)$-algebra $NPow^*(X)$ whose operation

$$\delta^{NPow^*(X)} : T \to (T^*)^X$$

is defined as follows:

For all $\{n_x \mid x \in X\} \subseteq \mathbb{N}$, $t = ()\{(x,i) \to t_{x,i} \mid x \in X,\ 1 \le i \le n_x\} \in T$ and $x \in X$,

$$\delta^{NPow^*(X)}(t)(x) \;=\; (t_{x,1}, \ldots, t_{x,n_x}).$$

9.7 Product algebras

Let $\Sigma = (S, F)$ be a signature and $\mathcal{A} = (\mathcal{A}_i)_{i \in I}$ be a tuple of Σ-algebras.

The product of $\mathcal{A}$ in Alg_Σ is given by the Σ-algebra $\mathcal{B} = \prod_{i \in I} \mathcal{A}_i$ that is defined as follows:

- For all $e \in \mathcal{T}(S)$, $\mathcal{B}(e) = \times_{i \in I} \mathcal{A}_i(e)$.
- For all $f : e \to e' \in F$, $f^{\mathcal{B}} = \prod_{i \in I} f^{\mathcal{A}_i} : \mathcal{B}(e) \to \mathcal{B}(e')$.

For all $i \in I$, the projection $\pi_i =_{def} (\pi_{i,e} : \mathcal{B}(e) \to \mathcal{A}_i(e))_{e \in \mathcal{T}(S)}$ is Σ-homomorphic: For all $f : e \to e' \in F$,

$$\pi_{i,e'} \circ f^{\mathcal{B}} = \pi_{i,e'} \circ \prod_{i \in I} f^{\mathcal{A}_i} \overset{(7)\ in\ chapter\ 2}{=} f^{\mathcal{A}_i} \circ \pi_{i,e}.$$

Let $\mathcal{C}$ be a Σ-algebra and $(h_i : \mathcal{C} \to \mathcal{A}_i)_{i \in I}$ be a tuple of Σ-homomorphisms. Then $\langle h_i \rangle_{i \in I} =_{def} (\langle h_{i,e} \rangle_{i \in I} : \mathcal{C}(e) \to \mathcal{B}(e))_{e \in \mathcal{T}_{fo}(S)}$ is also Σ-homomorphic:

For all $i \in I$ and $f : e \to e' \in F$,

$$(\langle h_i \rangle_{i\in I})_{e'} \circ f^{\mathcal{C}} = \langle h_{i,e'} \rangle_{i\in I} \circ f^{\mathcal{C}} \stackrel{(6)\ in\ chapter\ 2}{=} \langle h_{i,e'} \circ f^{\mathcal{C}} \rangle_{i\in I} \stackrel{h_i\ is\ \Sigma-hom.}{=} \langle f^{\mathcal{A}_i} \circ h_{i,e} \rangle_{i\in I}$$

$$= \langle f^{\mathcal{A}_i} \circ \pi_{i,e} \circ \langle h_{i,e} \rangle_{i\in I} \rangle_{i\in I} \stackrel{(6)\ in\ chapter\ 2}{=} \langle f^{\mathcal{A}_i} \circ \pi_{i,e} \rangle_{i\in I} \circ \langle h_{i,e} \rangle_{i\in I} = \prod_{i\in I} f^{\mathcal{A}_i} \circ \langle h_{i,e} \rangle_{i\in I}$$
$$\stackrel{Def.\ f^{\mathcal{B}}}{=} f^{\mathcal{B}} \circ \langle h_{i,e} \rangle_{i\in I} = f^{\mathcal{B}} \circ (\langle h_i \rangle_{i\in I})_e.$$

Uniqueness of $\langle h_i \rangle_{i\in I}$ follows from uniqueness of $\langle h_i \rangle_{i\in I}$ as a $\mathcal{T}_{fo}(S)$-sorted function.

Example 9.3 $\prod Dyn$

Let $\mathcal{A}$ be a $coStream(X)$-algebra with carrier Q and Y be a set (see section 8.2).

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 $f : Y \to Q$, $g : Q^Y \to Q$ and $x \in X$,

$$cons^{\mathcal{B}}(x,g)(f) = \pi_f(cons^{\mathcal{B}}(x,g)) = \pi_f(\langle cons^{\mathcal{A}} \circ \pi_{f,X\times state} \rangle_{f:Y\to Q}(x,g))$$
$$= cons^{\mathcal{A}}(\pi_{f,X\times state}(x,g)) = cons^{\mathcal{A}}(x, \pi_f(g)) = cons^{\mathcal{A}}(g(f), x).$$

- For all $y \in Y$ and $f : Y \to Q$, $\alpha^{\mathcal{B}}(y)(f) = f(y)$.

[20], Def. 4, provides this product automaton for the case $Y = 1$.

Given a semiring R, [161], section 3 defines a *weighted version* of $\mathcal{B}$ for $Y = 1$:

Let $\mathcal{A}$ be a $WcoStream(X,R)$-algebra (see section 8.2) with carrier Q and

$$(\delta^{\mathcal{A}})^* : X \times R_\omega^Q \to R_\omega^Q$$

be the $SMod_R$-extension of $\delta^{\mathcal{A}}$ (see chapter 19). 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 $q \in Q$, $g : Q \to R_\omega^Q$ and $x \in X$,

$$\begin{array}{l} cons^{\mathcal{B}}(x,g)(q) = \pi_q(cons^{\mathcal{B}}(x,g)) = \pi_q(\langle cons^{\mathcal{A}^*} \circ \pi_{q,X\times state}\rangle_{q\in Q}(x,g)) \\ = cons^{\mathcal{A}^*}(\pi_{q,X\times state}(x,g)) = \delta^{\mathcal{A}^*}(\pi_q(g),x) = \delta^{\mathcal{A}^*}(g(q),x). \end{array}$$

- For all $q \in Q$, $\alpha^{\mathcal{B}}(\epsilon)(q) = 1 \cdot q$ (see chapter 2). ❑

9.8 Sum algebras

Let $\Sigma = (S, F)$ be a signature and $\mathcal{A} = (\mathcal{A}_i)_{i\in I}$ be a tuple of Σ-algebras.

The sum of $\mathcal{A}$ in Alg_Σ is given by the Σ-algebra $\mathcal{B} = \coprod_{i\in I} \mathcal{A}_i$ that is defined as follows (see, e.g., [156], section 4.1; [139], section 2.2.1; or [81], Proposition 2.1.5):

- For all $e \in \mathcal{T}(S)$, $\mathcal{B}(e) = \biguplus_{i\in I} \mathcal{A}_i(e)$.
- For all $f : e \to e' \in F$, $f^{\mathcal{B}} = \coprod_{i\in I} f^{\mathcal{A}_i} : \mathcal{B}(e) \to \mathcal{B}(e')$.

For all $i \in I$, the injection $\iota_i = (\iota_{i,e} : \mathcal{A}_i(e) \to \mathcal{B}(e))_{e\in\mathcal{T}(S)}$ is Σ-homomorphic: For all $f : e \to e' \in F$,

$$f^{\mathcal{B}} \circ \iota_{i,e} = \coprod_{i\in I} f^{\mathcal{A}_i} \circ \iota_{i,e} \overset{(18)\ in\ chapter\ 2}{=} \iota_{i,e'} \circ f^{\mathcal{A}_i}.$$

Let $\mathcal{C}$ be a Σ-algebra and $(h_i : \mathcal{A}_i \to \mathcal{C})_{i\in I}$ be a tuple of Σ-homomorphisms. Then $[h_i]_{i\in I} = ([h_{i,e}]_{i\in I} : \mathcal{C}(e) \to \mathcal{B}(e))_{e\in\mathcal{T}_{fo}(S)}$ is also Σ-homomorphic:

For all $i \in I$ and $f : e \to e' \in F$,

$$\begin{aligned}([h_i]_{i\in I})_{e'} \circ f^{\mathcal{B}} &= [h_{i,e'}]_{i\in I} \circ f^{\mathcal{B}} \overset{Def.\ f^{\mathcal{B}}}{=} [h_{i,e'}]_{i\in I} \circ \textstyle\coprod_{i\in I} f^{\mathcal{A}_i} = [h_{i,e'}]_{i\in I} \circ [\iota_{i,e'} \circ f^{\mathcal{A}_i}]_{i\in I} \\ &\overset{(17)\ in\ chapter\ 2}{=} [[h_{i,e'}]_{i\in I} \circ \iota_{i,e'} \circ f^{\mathcal{A}_i}]_{i\in I} = [h_{i,e'} \circ f^{\mathcal{A}_i}]_{i\in I} \overset{h_i\ is\ \Sigma-hom.}{=} [f^{\mathcal{C}} \circ h_{i,e}]_{i\in I} \\ &\overset{(17)\ in\ chapter\ 2}{=} f^{\mathcal{C}} \circ [h_{i,e}]_{i\in I} = f^{\mathcal{C}} \circ ([h_i]_{i\in I})_e.\end{aligned}$$

Uniqueness of $[h_i]_{i\in I}$ follows from uniqueness of $[h_i]_{i\in I}$ as a $\mathcal{T}_{fo}(S)$-sorted function.

Example 9.4 $\coprod DAut$

Let $\mathcal{A}$ be a $Med(X)$-algebra (see section 8.3) 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 $q \in Q$, $f : Q \to Y$ and $x \in X$,
$$\delta^{\mathcal{B}}(q,f)(x) = [\iota_{f,state^X} \circ \delta^{\mathcal{A}}]_{f:Q\to Y}(q,f)(x) = [\iota_{f,state^X} \circ \delta^{\mathcal{A}}]_{f:Q\to Y}(\iota_f(q))(x)$$
$$= \iota_{f,state^X}(\delta^{\mathcal{A}}(q))(x) = \iota^X_{f,state}(\delta^{\mathcal{A}}(q))(x) = \iota_f(\delta^{\mathcal{A}}(q)(x)) = (\delta^{\mathcal{A}}(q)(x), f).$$
- For all $q \in Q$, and $f : Q \to Y$, $\beta^{\mathcal{B}}(q,f) = f(q)$.

[20], Def. 5, provides this sum automaton for the case $Y = 2$.

Given a semiring R, [161], 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
$$(\delta^{\mathcal{A}})^* : R^Q_\omega \to (R^Q_\omega)^X$$

be the $SMod_R$-extension of $\delta^{\mathcal{A}}$ (see chapter 19). Then $\mathcal{A}^*$ with $\mathcal{A}^*(state) = R^Q_\omega$ 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:

- $\mathcal{B}(state) = R^Q_\omega \times R^Q$.
- For all $f : Q \to R$, $g \in R^Q_R$ and $x \in X$,

$$\delta^{\mathcal{B}}(g,f)(x) = [\iota_{f,state^X} \circ \delta^{\mathcal{A}^*}]_{f:Q\to R}(q,g)(x) = [\iota_{f,state^X} \circ \delta^{\mathcal{A}^*}]_{f:Q\to R}(\iota_f(g))(x)$$
$$= \iota_{f,state^X}(\delta^{\mathcal{A}^*}(g))(x) = \iota^X_{f,state}(\delta^{\mathcal{A}^*}(g))(x) = \iota_f(\delta^{\mathcal{A}^*}(g)(x)) = (\delta^{\mathcal{A}^*}(g)(x), f).$$

- For all $f : Q \to R$ and $g \in R^Q_R$, $\beta^{\mathcal{B}}(g,f) = f^*(g)$ where $f^* : R^Q_\omega \to R$ is the $SMod_R$-extension of f (see chapter 19). ❏

Let $\Sigma = (S,F)$ be a signature and $\mathcal{A}$ be a Σ-algebra with carrier A.

9.9 Invariant algebras

An S-sorted subset B of A is a **Σ-invariant** of $\mathcal{A}$ if for all $f : e \to e' \in F$ and $a \in B_e$, $f^{\mathcal{A}}(a) \in B_{e'}$.

If Σ is constructive, then $\lambda s.\emptyset$ is the least Σ-invariant of $\mathcal{A}$ and $\lambda s.A_s$ is the greatest Σ-invariant of $\mathcal{A}$.

Given $a \in A$, the least invariant of $\mathcal{A}$ that contains a is denoted by $\langle a\rangle$. Its elements are also called **Σ-derivatives of** a.

For the construction of $\langle a\rangle$ if Σ is destructive, see Theorem 9.13 (4).

A Σ-invariant B of $\mathcal{A}$ induces the **Σ-subalgebra** $\mathcal{A}|_B$ of $\mathcal{A}$:

- For all $e \in \mathcal{T}_{fo}(S)$, $(\mathcal{A}|_B)(e) =_{def} B_e$.
- For all $f : e \to e' \in F$ and $a \in B_e$, $f^{\mathcal{A}|_B}(a) =_{def} f^{\mathcal{A}}(a)$.

Hence the inclusion map $inc_B : B \to A$ that sends $a \in B$ to itself is a Σ-homomorphism from $\mathcal{A}|_B$ to $\mathcal{A}$.

Σ-invariants of $\mathcal{A}$ provide the tool for building **restrictions** of the model given by $\mathcal{A}$:

If B consists of all elements of A satisfying a given constraint, then $\mathcal{A}|_B$ equips the constraint with a suitable algebraic structure.

9.10 Quotient algebras

Let $\mathcal{A}, \mathcal{B}$ be Σ-algebras with carriers A resp. B.

An S-sorted relation $R \subseteq A \times B$ is a **Σ-bisimulation** if for all $f : e \to e' \in F$ 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 Σ-bisimulation **on $\mathcal{A}$**.

A Σ-bisimulation on $\mathcal{A}$ is called a **Σ-congruence** if it is an equivalence relation.

If Σ is constructive, then $\lambda s.\Delta^2_{A_s}$ is the least Σ-congruence on $\mathcal{A}$ and $\lambda_s.A^2_s$ is the greatest Σ-congruence on $\mathcal{A}$.

A Σ-congruence R on $\mathcal{A}$ induces the **Σ-quotient (algebra) $\mathcal{A}/R$ of $\mathcal{A}$ by R**:

- For all $e \in \mathcal{T}_{fo}(S)$, $(\mathcal{A}/R)(e) =_{def} A_e/R_e$.
- For all $f : e \to e' \in F$ and $a \in A_e$, $f^{\mathcal{A}/R}([a]_R) =_{def} [f^{\mathcal{A}}(a)]_R$.

Hence the natural map $nat_R : A \to A/R$ that sends $a \in A$ to the equivalence class $[a]_R$ is a Σ-homomorphism from $\mathcal{A}$ to $\mathcal{A}/R$.

Σ-congruences of $\mathcal{A}$ provide the tool for building **abstractions** of the model given by $\mathcal{A}$:

If R consists of all pairs $(a,b) \in A^2$ such that a and b are to be considered equivalent, then $\mathcal{A}/R$ equips the equivalences with a suitable algebraic structure.

Lemma 9.5

Let R be a Σ-congruence. For all $f : e \to e' \in Arr_\Sigma$ and $(a, b) \in R$, $(f^{\mathcal{A}}(a), f^{\mathcal{A}}(b)) \in R$.

Proof.

$$nat(f^{\mathcal{A}}(a)) \overset{Lemma\ 9.2}{=} f^{\mathcal{A}/R}(nat(a)) = f^{\mathcal{A}/R}([a]_R) = f^{\mathcal{A}/R}([b]_R)$$
$$= f^{\mathcal{A}/R}(nat(b)) \overset{Lemma\ 9.2}{=} nat(f^{\mathcal{A}}(b)),$$

i.e., $(f^{\mathcal{A}}(a), f^{\mathcal{A}}(b)) \in R$. ❏

Theorem 9.6

(1) Let $\sim$ be a Σ-bisimulation on A. Then the equivalence closure $\sim^{eq}$ of $\sim$ (see chapter 3) is a Σ-congruence.

(2) The *greatest* Σ-bisimulation on A, called **Σ-bisimilarity**, is an equivalence relation and thus agrees with the greatest Σ-congruence on A.

(3) A $Mod(S)$-morphism $h : A \to B$ is a Σ-homomorphism from $\mathcal{A}$ to $\mathcal{B}$ iff the S-sorted relation

$$graph(h) =_{def} (\{(a, h_s(a) \mid a \in A_s\})_{s \in S}$$

is a Σ-bisimulation.

Proof. Given an S-sorted binary relation $\sim$ on A, $\sim^{eq}$ is the least fixpoint of

$$\begin{array}{rcl}\Phi(\sim) : \mathsf{X}_{e\in\mathcal{T}_{fo}(S)}\mathcal{P}(A_e^2) & \to & \mathsf{X}_{e\in\mathcal{T}_{fo}(S)}\mathcal{P}(A_e^2) \\ R & \mapsto & (\sim_e \cup \Delta^2_{A_e} \cup R_e^{-1} \cup R_e \cdot R_e)_{e\in\mathcal{T}_{fo}(S)}.\end{array}$$

Moreover, let $sucs(\sim)$ be the $\mathcal{T}_{fo}(S)$-sorted set defined by

$$sucs(\sim)_e = \{(a,b) \in A_e^2 \mid \forall\ f : e \to e' \in F : f^A(a) \sim_{e'} f^A(b)\}$$

for all $e \in \mathcal{T}_{fo}(S)$.

(1) Since $\sim^{eq}$ is an equivalence relation, it remains to show that $\sim^{eq}$ is a Σ-bisimulation, which holds true iff

$$\sim^{eq} \subseteq sucs(\sim^{eq}). \tag{3}$$

Since $\sim^{eq} = lfp(\Phi(\sim))$, fixpoint induction (see chapter 3) implies (3) if $sucs(\sim)$ is $\Phi(\sim)$-closed, i.e., if $\Phi(\sim)(sucs(\sim)) \subseteq sucs(\sim)$.

So let $(a,b) \in \Phi(\sim)(sucs(\sim^{eq}))$. Hence $(a,b) \in sucs(\sim^{eq})$ or we have one of the following three cases:

Case 1: $a = b$. Then for all $f : e \to e' \in F$, $f^A(a) = f^A(b)$ and thus $f^A(a) \sim^{eq} f^A(b)$ because $\sim^{eq}$ is reflexive.

Case 2: $b \sim^{eq} a$. Then for all $f : e \to e' \in F$, $f^A(b) \sim^{eq} f^A(a)$. 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 $f : e \to e' \in F$,

$$f^A(a) \sim^{eq} f^A(c) \quad \text{and} \quad f^A(c) \sim^{eq} f^A(b).$$

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 $sucs(\sim^{eq})$. Therefore, $sucs(\sim^{eq})$ is $\Phi(\sim)$-closed.

(2) Let R_Σ be the greatest Σ-bisimulation on $\mathcal{A}$. Suppose that

$$R_\Sigma^{eq} \text{ is a } \Sigma\text{-bisimulation.} \tag{4}$$

Since R_Σ is the *greatest* Σ-bisimulation, (4) implies $R_\Sigma^{eq} \subseteq R_\Sigma$. Since R_Σ is a subset of R_Σ^{eq}, we conclude that both relations agree with each other. Hence R_Σ is an equivalence relation and thus a Σ-congruence. It remains to show (4), which holds true iff

$$R_\Sigma^{eq} \subseteq sucs(R_\Sigma^{eq}). \tag{5}$$

Since $R_\Sigma^{eq} = lfp(\Phi(R_\Sigma))$, fixpoint induction (see chapter 3) implies (5) if $sucs(R_\Sigma^{eq})$ is $\Phi(R_\Sigma)$-closed, i.e., if $\Phi(R_\Sigma)(sucs(R_\Sigma^{eq})) \subseteq sucs(R_\Sigma^{eq})$.

So let $(a,b) \in \Phi(R_\Sigma)(sucs(R_\Sigma^{eq}))$. Hence $(a,b) \in sucs(R_\Sigma^{eq})$ or we have one of the following three cases:

Case 1: $a = b$. Then for all $f : e \to e' \in F$, $f^A(a) = f^A(b)$ and thus $(f^A(a), f^A(b)) \in R_\Sigma^{eq}$ because R_Σ^{eq} is reflexive.

Case 2: $(b,a) \in R_\Sigma^{eq}$. Then for all $f : e \to e' \in F$, $(f^A(b), f^A(a)) \in R_\Sigma^{eq}$. Hence $(f^A(a), f^A(b)) \in R_\Sigma^{eq}$ because R_Σ^{eq} is symmetric.

Case 3: $(a,c), (c,b) \in R_\Sigma^{eq}$ for some $c \in A$. Then for all $f : e \to e' \in F$,

$$(f^A(a), f^A(c)), (f^A(c), f^A(b)) \in R_\Sigma^{eq}.$$

Hence $(f^A(a), f^A(c)) \in R_\Sigma^{eq}$ because R_Σ^{eq} is transitive.

We conclude that in all three cases, (a,b) belongs to $sucs(R_\Sigma^{eq})$. Therefore, $sucs(R_\Sigma^{eq})$ is $\Phi(R_\Sigma)$-closed.

(3) "$\Rightarrow$": Let h be Σ-homomorphic, $e \in \mathcal{T}_{fo}(S)$, $(a,b) \in graph(h)_e$ and $f : e \to e' \in F$. Then $h(a) = b$ and thus $h(f^{\mathcal{A}}(a)) = f^{\mathcal{B}}(h(a)) = f^{\mathcal{B}}(b)$, i.e., $(f^{\mathcal{A}}(a), f^{\mathcal{B}}(b)) \in graph(h)_{e'}$.

"$\Leftarrow$": Let $graph(h)$ be a Σ-bisimulation, $f : e \to e' \in F$ and $a \in A_e$. Then

$$(a, h(a)), (f^{\mathcal{A}}(a), h(f^{\mathcal{A}}(a))) \in graph(h)$$

and thus $(f^{\mathcal{A}}(a), f^{\mathcal{B}}(h(a))) \in graph(h)$. Hence $h(f^{\mathcal{A}}(a)) = f^{\mathcal{B}}(h(a))$. ❑

9.11 Term folding

Let $\Sigma = (S, C)$ be a constructive polynomial signature, $V \in Set^S$ and $\mathcal{A}$ be a Σ-algebra with carrier A.

A **term valuation of** V **in** A is an S-sorted function $g : V \to A$.

From now on, A^V denotes the set of term valuations of V in A.

Given $g \in A^V$, the **term extension** $g^* : T_\Sigma(V) \to A$ **of** g **to** $T_\Sigma(V)$, also called **term folding**, is the $\mathcal{T}_{po}(S)$-sorted function that is defined inductively as follows:

- For all $I \subseteq \mathcal{I}$, $g^*_I = id_I$.
- For all $s \in S$ and $x \in V_s$, $g^*_s(x) = g_s(x)$.
- For all $c : e \to s \in \mathcal{I} \cup C$ and $t \in T_\Sigma(V)_e$,

$$g^*_s(c(t)) = c^{\mathcal{A}}(g^*_e(t)). \quad (1)$$

- For all $e = (e_i)_{i \in I} \in \mathcal{T}_{po}(S)^I$, $t = (t_i)_{i \in I} \in \times_{i \in I} T_\Sigma(V)_{e_i}$ and $i \in I$,

$$\pi_i(g^*_e(t)) = g^*_{e_i}(t_i). \quad (2)$$

- For all $e = (e_i)_{i \in I} \in \mathcal{T}_{po}(S)^I$, $i \in I$ and $t \in T_\Sigma(V)_{e_i}$,

$$g_e^*(i(t)) = \iota_i(g_{e_i}^*(t)). \tag{3}$$

Intuitively, g^* **evaluates** $t \in T_\Sigma(V)$ in $\mathcal{A}$ and thus computes a **denotational semantics** of t [78].

In particular, $id_A^* : T_\Sigma(A) \to \mathcal{A}$ 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 $c : e \to s \in C$ and $a \in A_e$, $id_A^*(c(a)) = c^{\mathcal{A}}(a)$.

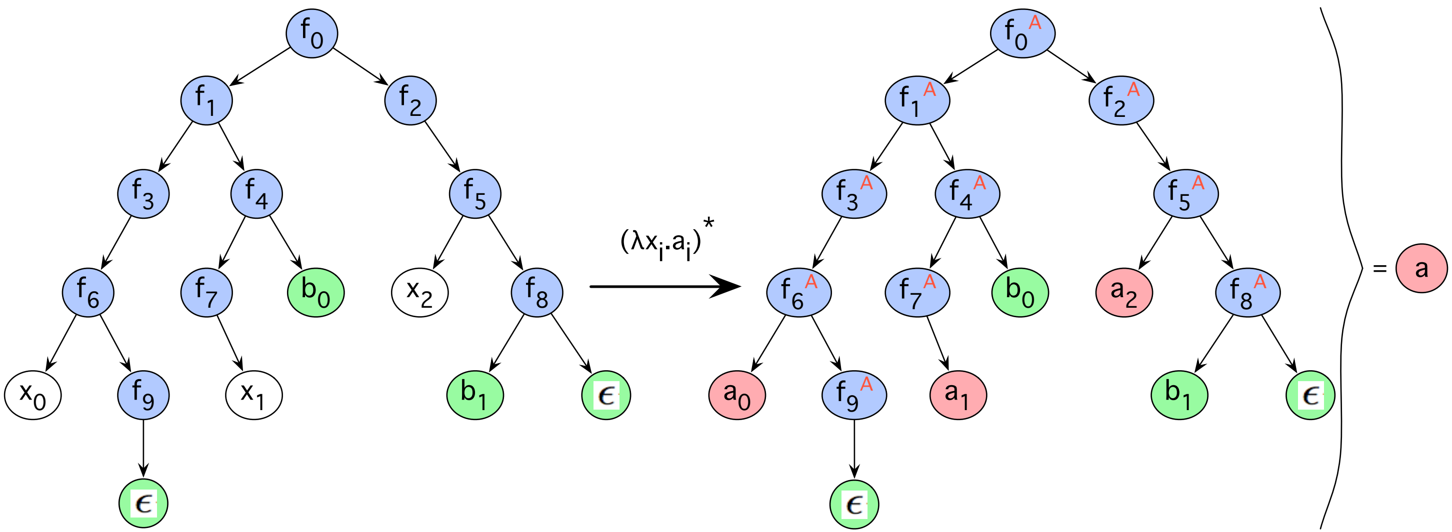


Illustration of term folding

Given $e \in \mathcal{T}_{po}(S)$ and $t, t' \in T_\Sigma(V)_e$, $g \in A^V$ **solves** the (**first-order**) **Σ-equation** $t = t'$ in $\mathcal{A}$, written as $\mathcal{A} \models_g t = t'$, if $g^*(t) = g^*(t')$.

$\mathcal{A}$ **satisfies** the (**first-order**) **conditional Σ-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 abbreviated to *True* and mostly omitted, i.e., a conditional equation $\mathit{True} \Rightarrow t = t'$ is simply written as $t = t'$.

Consequently, $\mathcal{A}$ satisfies $t = t'$ if all $g \in A^V$ solve $t = t'$.

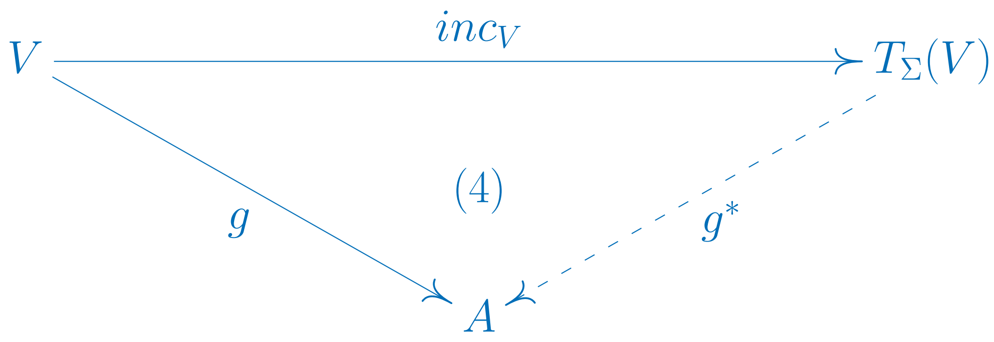
If $\mathcal{A} = T_\Sigma(V')$ for some $V' \in \mathit{Set}^S$, then g is called a **substitution** because

$$g^* : T_\Sigma(V) \to T_\Sigma(V')$$

replaces the variables of a term by terms: For all $x \in V$, x is replaced by $g(x)$.

Theorem 9.7

Let $V \in Set^S$. $T_\Sigma(V)$ is a **free Σ-algebra over** V, i.e., for all Σ-algebras $\mathcal{A}$ with carrier A and $g \in A^V$, g^* is the only Σ-homomorphism from $T_\Sigma(V)$ to $\mathcal{A}$ that satisfies (4).



In particular, if for all $s \in S$, $V_s = \emptyset$, then there is exactly one term valuation $g \in A^V$, (4) reduces to the uniqueness of g^* and thus $T_\Sigma = T_\Sigma(V)$ is initial in Alg_Σ. g^* no longer depends on g and is denoted by $fold^{\mathcal{A}}$. This notation is also used for the restriction of g^* to T_Σ if V is nonempty.

Proof. By (1), g^* satisfies (4).

g^* is Σ-homomorphic: For all $c : e \to s \in C$ and $t \in T_\Sigma(V)_e$,

$$g^*(c^{T_\Sigma(V)}(t)) = g^*(c(t)) \overset{(1)}{=} c^{\mathcal{A}}(g^*(t)).$$

g^* is unique: Let $h : T_\Sigma(V) \to \mathcal{A}$ be a Σ-homomorphism with $h \circ inc_V = g$.

- For all $x \in V$, $g^*(x) = g(x) \stackrel{h \circ inc_V = g}{=} h(x)$.
- For all $c : e \to s \in C$ and $t \in T_\Sigma(V)_e$,

$$g^*(c(t)) \stackrel{(1)}{=} c^{\mathcal{A}}(g^*(t)) \stackrel{ind.\ hyp.}{=} c^{\mathcal{A}}(h(t)) \stackrel{h\ hom.}{=} h(c^{T_\Sigma(V)}(t)) = h(c(t)).$$

- For all $(e_i)_{i \in I} \in \mathcal{T}_{po}(S)^I$, $t = (t_i)_{i \in I} \in \times_{i \in I} T_\Sigma(V)_{e_i}$ and $i \in I$,

$$\pi_i(g^*(t) \stackrel{(2)}{=} g^*(t_i) \stackrel{ind.\ hyp.}{=} h(t_i) = h(\pi_i(t)) = \pi_i(h(t)).$$

- For all $(e_i)_{i \in I} \in \mathcal{T}_{po}(S)^I$, $i \in I$ and $t \in T_\Sigma(V)_{e_i}$,

$$g^*(i(t)) \stackrel{(3)}{=} \iota_i(g^*(t)) \stackrel{ind.\ hyp.}{=} \iota_i(h(t)) = h(\iota_i(t)) = h(i(t)).$$

Hence $g^* = h$. ❑

Exercise 14 Show $fold^{Bool} = \beta^{Bro(X)}$.

Exercise 15 Show by structural induction that for all $t \in T_{Reg(X)}$,

$$\epsilon \in fold^{Lang(X)}(t) \quad \Leftrightarrow \quad fold^{Bool}(t) = 1.$$

A Σ-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_Σ whose objects are all reachable Σ-algebras.

By Lemma 13.1 (2), for all $\mathcal{A} \in RAlg_\Sigma$, $\mathcal{A} \cong T_\Sigma/ker(fold^{\mathcal{A}})$.

Lemma 9.8

Let $\mathcal{K}$ be a full subcategory of $RAlg_\Sigma$ and the Σ-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 equation 2.2.9) and $\mathcal{B}(s) = T_{\Sigma,s}/R_s$.
- For all $c : e \to s \in C$ 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 Σ-homomorphism from $\mathcal{B}$ to $\mathcal{A}$.
In particular, $\mathcal{B}$ is initial in $RAlg_\Sigma$.

Proof. By Lemma 13.1 (2), there is a unique Σ-monomorphism $h : \mathcal{B} \to \prod\mathcal{K}$ such that $h \circ nat_R = \langle fold^{\mathcal{A}}\rangle_{\mathcal{A}\in\mathcal{K}}$. Hence for all $\mathcal{A} \in \mathcal{K}$, $\pi_{\mathcal{A}} \circ h : \mathcal{B} \to \mathcal{A}$ is Σ-homomorphic. Suppose that there are two Σ-homomorphisms $h_1, h_2 : \mathcal{B} \to \mathcal{A}$.

Since T_Σ is initial in Alg_Σ, $h_1 \circ nat_R = h_2 \circ nat_R$. 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 Σ-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 [161], section 5, and a quotient of the initial reachable Σ-algebra. ❑

Lemma 9.9 (Substitutionslemma)

For all Σ-algebras $\mathcal{A}$ with carrier A, $g \in A^V$ and Σ-homomorphisms $h : \mathcal{A} \to \mathcal{B}$,

$$(h \circ g)^* = h \circ g^*.$$

Proof. Since $h \circ g^* \circ inc_V = h \circ g$, the conjecture follows from the fact that $(h \circ g)^*$ is the only Σ-homomorphism $h' : T_\Sigma(V) \to \mathcal{B}$ with $h' \circ inc_V = h \circ g$. ❑

Since g^* is the only Σ-homomorphism from $T_\Sigma(V)$ to $\mathcal{A}$ that satisfies (5), $fold^{\mathcal{A}}$ is the only Σ-homomorphism from T_Σ to $\mathcal{A}$, i.e., **T_Σ is initial in Alg_Σ**.

9.12 Term grounding

Let $V \in \mathcal{I}$.

$$\Sigma(V) = (S, C \cup \{val_s : V_s \to s \mid s \in S\})$$

is called the **grounding of Σ on V**.

$T_\Sigma(V)$ is a $\Sigma(V)$-algebra: For all $s \in S$ and $x \in V_s$, $val_s^{T_\Sigma(V)}(x) =_{def} val_s(x)$.

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 $h, h' : T_\Sigma(V) \to \mathcal{A}$ 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 Σ-homomorphic, we conclude $h = h'$.

Therefore, **$T_\Sigma(V)$ 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 ϵ 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 chapter 15).

9.13 Sample initial algebras

Since initial algebras are unique up to isomorphism,

- $T_{Nat} \cong \mathbb{N}$ is initial in Alg_{Nat} (see sample algebra 9.6.1), (1)
- $T_{Dyn(X,Y)} \cong Seq(X,Y)$ is initial in $Alg_{Dyn(X,Y)}$ (see sample algebra 9.6.3), (2)
- $T_{List(X)} \cong X^*$ is initial in $Alg_{List(X)}$ (see sample algebra 9.6.3), (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 9.6.10), (4)
- $T_{Tree(X)} \cong FTree(X)$ is initial in $Alg_{Tree(X)}$ (see sample algebra 9.6.13), (5)
- $T_{Reg(X)}$ is initial in $Alg_{Reg(X)}$. (6)

Given a constructive signature Σ and a Σ-algebra $\mathcal{A}$ with carrier A, $fold^{\mathcal{A}}$ denotes the unique Σ-homomorphism not only from T_Σ to $\mathcal{A}$, but also from isomorphic representations of T_Σ like those listed above.

In cases (1)-(5), the respective inductive (!) definition of $h =_{def} fold^{\mathcal{A}}$ reads as follows:

1. Nat-algebra $\mathbb{N}$

$$\begin{array}{rcl} h : \mathbb{N} & \to & A \\ 0 & \mapsto & zero^{\mathcal{A}} \\ n+1 & \mapsto & succ^{\mathcal{A}}(h(n)) \end{array}$$

2. $Dyn(X,Y)$-algebra $Seq(X,Y)$

$$\begin{array}{rcl} h : X^* \times Y & \to & A \\ (\epsilon, y) & \mapsto & \alpha^{\mathcal{A}}(y) \\ (xw, y) & \mapsto & cons^{\mathcal{A}}(x, h(w,y)) \end{array}$$

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^*

$$\begin{array}{rcl} h : X^* & \to & A \\ \epsilon & \mapsto & \alpha^{\mathcal{A}} \\ x \cdot w & \mapsto & cons^{\mathcal{A}}(x, h(w)) \end{array}$$

h is $List(X)$-homomorphic:

$$\begin{array}{l} h(\alpha^{X^*}) = h(\epsilon) = \alpha^{\mathcal{A}}, \\ h(cons^{X^*}(x, w)) = h(x \cdot w) = cons^{\mathcal{A}}(x, h(w)). \end{array}$$

h is the only $List(X)$-homomorphism from X^* to $\mathcal{A}$:
Let $h' : X^* \to A$ be $List(X)$-homomorphic. Then

$$\begin{array}{l} h'(\epsilon) = h(\alpha^{X^*}) \overset{h'\ hom.}{=} \alpha^{\mathcal{A}} = h(\epsilon), \\ h'(x \cdot w) = h'(cons^{X^*}(x, w)) \overset{h'\ hom.}{=} cons^{\mathcal{A}}(x, h'(w)) \overset{ind.\ hyp.}{=} cons^{\mathcal{A}}(x, h(w)) = h(x \cdot w), \end{array}$$

i.e., $h' = h$.

Exercise 16 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 9.6.1).

4. $Bintree(X)$-algebra $FBin(X)$

$$\begin{aligned} h : ftr(2, X) &\rightarrow A \\ \Omega &\mapsto empty^{\mathcal{A}} \\ x\{0 \rightarrow t, 1 \rightarrow u\} &\mapsto bjoin^{\mathcal{A}}(x, h(t), h(u)) \end{aligned}$$

5. $Tree(X)$-algebra $FTree(X)$

$$\begin{aligned} h_{tree} : otr(\mathbb{N}, X) \cap ftr(\mathbb{N}, X) &\rightarrow A_{tree} \\ x &\mapsto join^{\mathcal{A}}(x, nil^{\mathcal{A}}) \\ x(t_1, \dots, t_n) &\mapsto join^{\mathcal{A}}(x, h_{trees}(t_1, \dots, t_n)) \\ h_{trees} : (otr(\mathbb{N}, X) \cap ftr(\mathbb{N}, X))^* &\rightarrow A_{trees} \\ \epsilon &\mapsto nil^{\mathcal{A}} \\ t \cdot ts &\mapsto cons^{\mathcal{A}}(h_{tree}(t), h_{trees}(ts)) \end{aligned}$$

6. $Reg(X)$-algebra $T_{Reg(X)}$

For all $t \in T_{Reg(X)}$, $fold^{Lang(X)}(t)$ (see sample algebra 9.6.19) is usually called the **language of** t that is accepted by, for instance, $(Bro(X), t)$ (see sample final algebra 9.18.11).

Exercise 17 Let *Bool* be the Σ-algebra of example 20 above. Given $t \in T_{\Sigma,state}$, show that the language of t contains ϵ iff $fold^{Bool}(t) = 1$. ❏

9.14 Context-free grammars and their models

A **context-free grammar** (**CFG**) $G = (S, X, R)$ consists of

- a finite set S of sorts, also called **nonterminals**,
- a set $X \subseteq \mathcal{I}$ of **terminals**,
- a finite S-sorted set $R = (R_s \subseteq \{s \to w \mid w \in (S \cup \mathcal{P}_+(X))^*\})_{s \in S}$ of **rules**.

Every $s \in S$ is supposed to be the left-hand side of at least one rule of R.

Let $S_X = S \cup \mathcal{P}_+(X)$ and $type : S_X^* \to \mathcal{T}_{po}(S)$ be inductively defined as follows:

- $type(\epsilon) = 1$.
- For all $s \in S_X$ and $w \in S_X^*$,

$$type(sw) \;=\; \begin{cases} type(w) & \text{if } s \in X, \\ s \times type(w) & \text{otherwise.} \end{cases}$$

The constructive signature

$$\Sigma(G) \;=\; (S, \{f_{s\to w} : type(w) \to s \mid s \to w \in R\})$$

is called the **abstract syntax of** G.

Why are the terminals of w removed from a rule $s \to w$ when it is turned into a constructor of $\Sigma(G)$? Because they do not contribute to the semantics of the product types created by $type$. E.g., for all $e, e' \in \mathcal{T}_{po}(S)$ and $x \in X$, $e \times x \times e'$ is equivalent to $e \times e'$ (see chapter 7).

However, the terminals of w may be used for naming the constructor for $s \to w$. For instance, the rule $s \to if\ 2\ then\ s\ else\ s$ yields the constructor $ite : 2 \times s \times s \to s$.

Finite ground $\Sigma(G)$-terms are called **syntax trees of** G.

The $\Sigma(G)$-algebra $Word(G)$, called the **word algebra of** G, recovers the concrete syntax from the abstract one. It is defined as follows:

- For all $s \in S$, $Word(G)_s = X^*$.
- For all $w_0 \dots w_n \in X^*$, $s_1, \dots, s_n \in S_X \setminus X$, $r = (s \to w_0 s_1 w_1 \dots s_n w_n) \in R$ and $(v_1, \dots, v_n) \in Word(G)_{s_1 \times \dots \times s_n}$,

$$f_r^{Word(G)}(v_1, \dots, v_n) = w_0 v_1 w_1 \dots v_n w_n.$$

Example 9.10 The grammar SAB consists of the sorts A, B, C, the terminals a and b and the rules

$$r_1 = C \to aB, \quad r_2 = C \to bA, \quad r_3 = C \to \epsilon,$$
$$r_4 = A \to aC, \quad r_5 = A \to bAA, \quad r_6 = B \to bC, \quad r_7 = B \to aBB.$$

Hence Σ(SAB) has the following constructors:

$$f_1 : B \to C, \quad f_2 : A \to C, \quad f_3 : 1 \to C,$$
$$f_4 : C \to A, \quad f_5 : A \times A \to A, \quad f_6 : C \to B, \quad f_7 : B \times B \to B.$$

The word algebra $\mathcal{W} = Word(SAB)$ reads as follows:

$$\mathcal{W}_A = \mathcal{W}_B = \mathcal{W}_C = \{a, b\}^*$$

and for all $v, w \in \{a,b\}^*$,

$$\begin{array}{lcccl} f_1^{\mathcal{W}}(w) & = & f_4^{\mathcal{W}}(w) & = & aw, \\ f_2^{\mathcal{W}}(w) & = & f_6^{\mathcal{W}}(w) & = & bw, \\ f_3^{\mathcal{W}} & = & \epsilon, & & \\ f_5^{\mathcal{W}}(v,w) & = & bvw, & & \\ f_7^{\mathcal{W}}(v,w) & = & avw. & & \end{array}$$ ❑

The **language** $L(G)$ **of** G is the S-sorted set of words over X that result from folding syntax trees in $Word(G)$, i.e.,

$$L(G) = img(fold^{Word(G)}).$$

Given an S-sorted set E of error messages, a **parser for** G is an S-sorted function

$$parse_G \;=\; (parse_{G,s} : X^* \to \mathcal{P}(T_{\Sigma(G),s}) + E)_{s \in S}$$

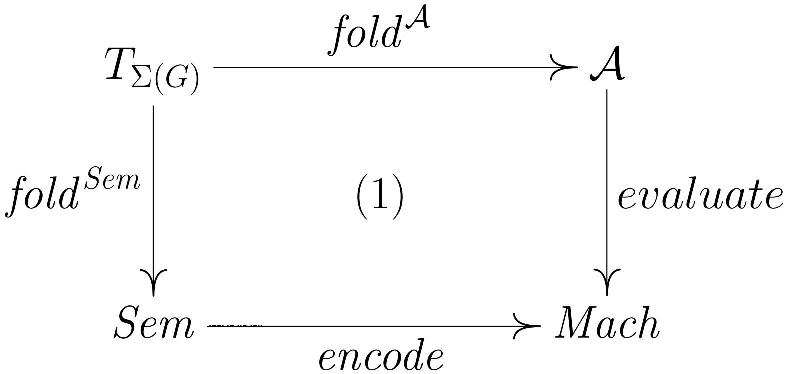
such that for all $w \in X^*$,

$$parse_G(w) \begin{cases} = (fold^{Word(G)})^{-1}(w) & \text{if } w \in L(G), \\ \in E & \text{otherwise.} \end{cases}$$

Every $\Sigma(G)$-algebra $\mathcal{A}$ may serve as a target language of a compiler of G: The **generic compiler for G induced by $parse_G$** is defined as the $(Alg_{\Sigma(G)} \times S)$-sorted function

$$compile_{parse_G} = ((\mathcal{P}(fold_s^{\mathcal{A}}) + id_E) \circ parse_G : X^* \to \mathcal{P}(\mathcal{A}(s) + E))_{(\mathcal{A},s) \in Alg_{\Sigma(G)} \times S}.$$

Whereas $\Sigma(G)$ describes the syntax of a *source* language, the carrier of $\mathcal{A}$ may consist of the programs of a *target* language. In this case, the correctness of a compiler to $\mathcal{A}$ may involve the compatibility of $\mathcal{A}$ with a further $\Sigma(G)$-algebra Sem (the *source model*), a further S-sorted set $Mach$ (the *target model* or *abstract machine*) and two S-sorted functions $evaluate : \mathcal{A} \to Mach$ and $encode : Sem \to Mach$ such that the following diagram commutes (see [113, 178]):



evaluate provides the interpreter of target programs in *Mach*, while *encode* expresses the source model in terms of the target model.

Due to the initiality of $T_{\Sigma(G)}$, (1) commutes if *encode* and *evaluate* are $\Sigma(G)$-homomorphic, which involves that *Mach* is $\Sigma(G)$-algebra. Since *Mach* is supposed to provide the semantics of the target programs given by the carrier A of $\mathcal{A}$, there may be a signature Σ' such that $T_{\Sigma'}$ concides with A such that each constructor of $\Sigma(G)$ is "implemented" by a Σ'-term and *evaluate* folds Σ'-terms in *Mach*. This implementation may determine a suitable definition of *encode* such that both *encode* and *evaluate* become $\Sigma(G)$-homomorphic. In this way, [178] shows the correctness of a compiler that translates imperative programs to flowcharts.

9.15 Removing left recursion

The one-step **derivation relation** $\rightarrow_G \subseteq (S_X^*)^2$ for a CFG $G = (S, X, R)$ is defined as follows:

$$\begin{array}{rcl} \rightarrow_G & = & \{(vsv', vwv') \mid s \rightarrow w \in R,\ v, v' \in S_X^*\} \cup \\ & & \{(v\overline{B}v', vxv') \mid B \subseteq X,\ x \in B\}. \end{array}$$

$\stackrel{+}{\rightarrow}_G$ and $\stackrel{*}{\rightarrow}_G$ denote the transitive resp. reflexive-transitive closure of $\rightarrow_G$.

For all $s \in S$, $L(G)_s = \{w \in X^* \mid s \stackrel{+}{\rightarrow}_G w\}$.

G is **left-recursive** if there are $s \in S$ and $w \in S_X^*$ with $s \stackrel{+}{\rightarrow}_G sw$.

Top-down parsers for a CFG $G = (S, X, R)$ proceed along $\to_G$ and thus may not terminate on some input words if G is left-recursive. Fortunately, the following procedure transforms G into a non-left-recursive grammar $G' = (S', X, R')$ such that $S \subseteq S'$ and for all $s \in S$, $L(G)_s = L(G')_s$:

- Repeat the following step as often as possible: For all pairs $(s \to s'v, s' \to w) \in R^2$ with $s \neq s'$ replace $s \to s'v$ by the new rule $s \to wv$. (1)
- Remove all rules of the form $s \to s$. (2)

(1) turns every derivation $s \xrightarrow{+}_G sw$ into a rule. Let R_0 be the set of rules after (1) and (2) have been performed. The non-left-recursive grammar G' is then defined as follows: Let $s \in S$.

$$
\begin{aligned}
S' &= S \cup \{s' \mid s \in recs(S)\}, \\
R' &= R_0 \setminus \{s \to w \in R_0 \mid s \in recs(S)\} \\
&\quad \cup \{s' \to ws' \mid s \to sw \in R_0\} \\
&\quad \cup \{s \to vs' \mid v \in nonrecs(s),\ s \in S\} \\
&\quad \cup \{s' \to \epsilon \mid s \in recs(S)\}
\end{aligned}
$$

where $recs(S) = \{s \in S \mid \exists\ s \to sw \in R_0\}$ and

$$nonrecs(s) = \{w \mid s \in recs(S),\ s \to w \in R_0,\ w \notin \{s\} \times S_X^*\}.$$

Example 9.11 The rules of a CFG for a subset of Java may read as follows:

$$
\begin{array}{lcl}
Commands & \to & Command\ Commands \mid Command \\
Command & \to & \{Commands\} \mid \overline{Ident} = Sum; \mid \\
 & & \texttt{if}\ Disjunct\ Command\ \texttt{else}\ Command \mid \\
 & & \texttt{if}\ Disjunct\ Command \mid \texttt{while}\ Disjunct\ Command \\
Sum & \to & Sum + Prod \mid Sum - Prod \mid Prod \qquad (1) \\
Prod & \to & Prod * Factor \mid Prod/Factor \mid Factor \qquad (2) \\
Factor & \to & \overline{\mathbb{Z}} \mid \overline{Ident} \mid (Sum) \\
Disjunct & \to & Conjunct \mathbin{||} Disjunct \mid Conjunct \\
Conjunct & \to & Literal\ \&\&\ Conjunct \mid Literal \\
Literal & \to & !Literal \mid Sum\ \overline{Rel}\ Sum \mid \overline{2} \mid (Disjunct)
\end{array}
$$

Rules with the same left-hand side are combined to a single one by summing up their right-hand sides with |. *Ident* and *Rel* denote given sets of identifiers ad binary relations, respectively. The rules' left-hand sides are the sorts of JavaLight, all other symbols except | that occur on right-hand sides and the elements of $\mathbb{Z}$, *Ident* and *Rel* form the set of terminals of JavaLight.

The use of three sorts for both arithmetic and Boolean expressions reflects the usual priorities of arithmetic resp. Boolean operators and thus allows the avoidance of superfluous brackets.

The above procedure adds to Javalight the sorts Sum' and $Prod'$ and replaces rules (1) amd (2) by the following ones:

$$\begin{array}{lcl} Sum & \to & Prod\ Sum' \qquad (3) \\ Sum' & \to & +Prod\ Sum' \mid -Prod\ Sum' \mid \epsilon \qquad (4) \\ Prod & \to & Factor\ Prod' \qquad (5) \\ Prod' & \to & *Factor\ Prod' \mid /Factor\ Prod' \mid \epsilon \qquad (6) \end{array}$$

The abstract syntax of the original Javalight consists of the sorts

$$Commands, Command, Sum, Prod, Factor, Disjunct, Conjunct, Literal, \overline{\mathbb{Z}}, \overline{Ident}, \overline{Rel}$$

and the constructors

$$\begin{array}{l} seq : Command \times Commands \to Commands, \\ embed : Command \to Commands, \\ block : Commands \to Command, \\ assign : String \times Sum \to Command, \end{array}$$

$$
\begin{aligned}
&cond : Disjunct \times Command \times Command \to Command,\\
&cond1, loop : Disjunct \times Command \to Command,\\
&sum : Prod \to Sum, && (7)\\
&plus, minus : Sum \times Prod \to Sum, && (8)\\
&prod : Factor \to Prod, && (9)\\
&times, div : Prod \times Factor \to Prod, && (10)\\
&embedI : \mathbb{Z} \to Factor,\\
&var : String \to Factor,\\
&encloseS : Sum \to Factor,\\
&disjunct : Conjunct \times Disjunct \to Disjunct,\\
&embedC : Conjunct \to Disjunct,\\
&conjunct : Literal \times Conjunct \to Conjunct,\\
&embedL : Literal \to Conjunct,\\
&not : Literal \to Literal,\\
&atom : Sum \times Rel \times Sum \to Literal,\\
&embedB : 2 \to Literal,\\
&encloseD : Disjunct \to Literal
\end{aligned}
$$

According to the replacement of rules (1) and (2) by (3)-(6), constructors (7)-(10) are exchanged with the following ones:

$$\begin{aligned}
&sum' : Prod \times Sum' \to Sum, && (11)\\
&plus', minus' : Prod \times Sum' \to Sum', && (12)\\
&nilS : 1 \to Sum', && (13)\\
&prod' : Factor \times Prod' \to Prod, && (13)\\
&times', div' : Factor \times Prod' \to Prod', && (14)\\
&nilP : 1 \to Prod'. && ❏
\end{aligned}$$

While the types of constructors (8)-(10) suggest that (8) and (10) will be interpreted as binary operatos and terms composed of these arrows will be evaluated from left to right, the intended meaning of (11)-(14) is less obvious. In fact, the new sorts Sum' and $Prod'$ represent sets of Haskell *sections* of the constructors (8) and (10). For instance, the section $(*5) : \mathbb{Z} \to \mathbb{Z}$ maps $x \in \mathbb{Z}$ to $x * 5$, and the left-associative evaluation of the term $x/y * z/z'$ becomes the left-to right application of the sections $(/y)$, $(*z)$ and $(/y)$ to x.

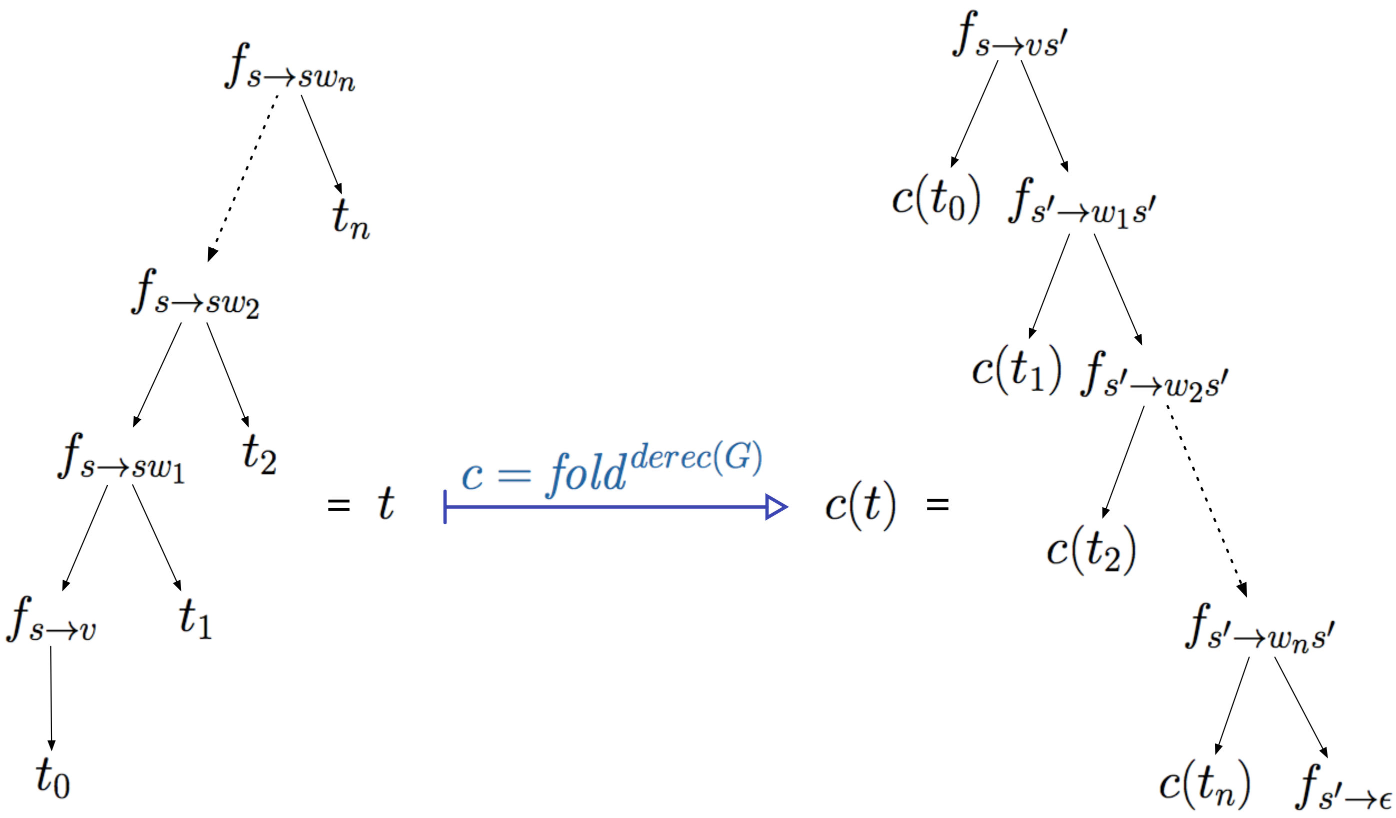
The step from a CFG $G = (S, X, R)$ to its non-left-recursive equivalent $G' = (S', X, R')$ modifies the abstract syntax and thus the syntax trees of G. The following $\Sigma(G)$-algebra $derec(G)$ interprets the constructors of $\Sigma(G)$ as functions on the syntax trees of G':

- For all $s \in S$, $derec(G)_s = T_{\Sigma(G'),s}$. (1)
- For all $s \in S \setminus recs(S)$ and $r \in R_0$, $f_{s\to v}^{derec(G)} = f_{s\to v}$. (2)
- For all $s \to sw \in R_0$, $v \in nonrecs(s)$, $t \in T_{\Sigma(G'),type(v)}$, $t' \in T_{\Sigma(G'),s'}$ and $u \in T_{\Sigma(G'),type(w)}$,

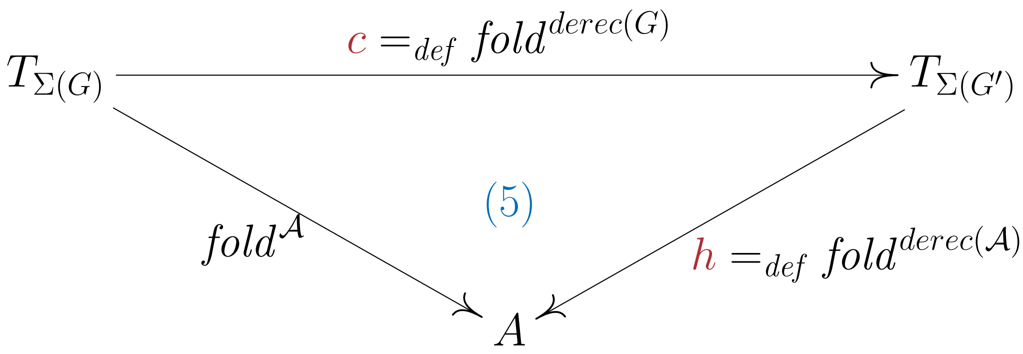
$$f_{s\to v}^{derec(G)}(t) = f_{s\to vs'}(t, f_{s'\to\epsilon}), \tag{3}$$

$$f_{s\to sw}^{derec(G)}(f_{s\to vs'}(t,t'), u) = f_{s\to vs'}(t, f_{s'\to ws'}(u,t')). \tag{4}$$

Folding a syntax tree of G in $derec(G)$ yields its G'-counterpart:



As any CFG G can be turned automatically into its left-non-recursive equivalent G', so every $\Sigma(G)$-algebra $\mathcal{A}$ can be transformed automatically into a $\Sigma(G')$-algebra $derec(\mathcal{A})$ such that folding syntax trees of G leads to to same results as folding their G'-counterparts in $derec(\mathcal{A})$:



Let A be the carrier of $\mathcal{A}$. $derec(\mathcal{A})$ is defined as follows:

- For all $s \in S$, $derec(\mathcal{A})(s) = A_s$.
- For all $s \in S \setminus recs(S)$ and $r \in R_0$, $f_r^{derec(\mathcal{A})} = f_r^{\mathcal{A}}$. (6)
- For all $s \to sw \in R_0$, $v \in nonrecs(s)$, $a \in A_{type(v)}$, $g : A_s \to A_s$, $b \in A_{type(w)}$ and $x \in A_s$,

$$\begin{aligned} derec(\mathcal{A})(s') &= A_s \to A_s, & (7)\\ f_{s' \to \epsilon}^{derec(\mathcal{A})}(\epsilon) &= id_{A_s}, & (8)\\ f_{s \to vs'}^{derec(\mathcal{A})}(a, g) &= g(f_{s \to v}^{\mathcal{A}}(a)), & (9)\\ f_{s' \to ws'}^{derec(\mathcal{A})}(b, g)(x) &= g(f_{s \to sw}^{\mathcal{A}}(x, b)). & (10) \end{aligned}$$

Proof of the commutativity of (5) by induction over the size of the sysntax trees of G.

Let $s \in S \setminus recs(S)$ und $t \in T_{\Sigma(G),s}$.

Then $t = f_r(u)$ for some $r = (s \to v) \in R_0$ und $u \in T_{\Sigma(G),type(v)}$. Hence

$$h(c(t)) = h(c(f_r(u))) \overset{c\ hom.}{=} h(f_r^{derec(G)}(c(u))) \overset{(2)}{=} h(f_r(c(u))) \overset{h\ hom.}{=} f_r^{derec(\mathcal{A})}(h(c(u)))$$
$$\overset{(6)}{=} f_r^{\mathcal{A}}(h(c(u))) \overset{ind.\ hyp.}{=} f_r^{\mathcal{A}}(fold^{\mathcal{A}}(u)) \overset{fold^{\mathcal{A}}\ hom.}{=} fold^{\mathcal{A}}(f_r(u)) = fold^{\mathcal{A}}(t).$$

Let $s \in recs(S)$ and $t \in T_{\Sigma(G),s}$. Then there are $v \in nonrecs(s)$, $t_0 \in T_{\Sigma(G),type(v)}$, $n \in \mathbb{N}$ and for all $1 \leq i \leq n$, $r_i = (s \to sw_i) \in R_0$ and $t_i \in T_{\Sigma(G),type(w_i)}$, such that

$$t = f_{r_n}(\dots(f_{r_1}(f_{s\to v}(t_0), t_1)\dots), t_n). \tag{11}$$

Hence

$$h(c(t)) \overset{(11)}{=} h(c(f_{r_n}(\dots(f_{r_1}(f_{s\to v}(t_0), t_1)\dots), t_n)))$$
$$\overset{c\ hom.}{=} h(f_{r_n}^{derec(G)}(\dots(f_{r_1}^{derec(G)}(f_{s\to v}^{derec(G)}(c(t_0))), c(t_1))\dots), c(t_n))$$
$$\overset{(3)}{=} h(f_{r_n}^{derec(G)}(\dots(f_{r_1}^{derec(G)}(f_{s\to vs'}(c(t_0), f_{s'\to\epsilon})), c(t_1))\dots), c(t_n)) \overset{(4)}{=} \dots$$
$$\overset{(4)}{=} h(f_{s\to vs'}(c(t_0), f_{r'_1}(c(t_1), \dots, f_{r'_n}(c(t_n), f_{s'\to\epsilon})\dots))$$
$$\overset{h\ hom.}{=} f_{s\to vs'}^{derec(\mathcal{A})}(h(c(t_0)), f_{r'_1}^{derec(\mathcal{A})}(h(c(t_1)), \dots, f_{r'_n}^{derec(\mathcal{A})}(h(c(t_n)), f_{s'\to\epsilon}^{derec(\mathcal{A})})\dots)$$
$$\overset{(8)}{=} f_{s\to vs'}^{derec(\mathcal{A})}(h(c(t_0)), f_{r'_1}^{derec(\mathcal{A})}(h(c(t_1)), \dots, f_{r'_n}^{derec(\mathcal{A})}(h(c(t_n)), id)\dots))$$

$$\stackrel{(9)}{=} f_{r_1'}^{derec(\mathcal{A})}(h(c(t_1)),\dots,f_{r_n'}^{derec(\mathcal{A})}(h(c(t_n)),id)\dots)(f_{s\to v}^{\mathcal{A}}(h(c(t_0)))) \stackrel{(10)}{=} \dots$$

$$\stackrel{(10)}{=} id(f_{r_n}^{\mathcal{A}}(\dots(f_{r_1}^{\mathcal{A}}(f_{s\to v}^{\mathcal{A}}(h(c(t_0))),h(c(t_1))))\dots,h(c(t_n))))$$

$$= f_{r_n}^{\mathcal{A}}(\dots(f_{r_1}^{\mathcal{A}}(f_{s\to v}^{\mathcal{A}}(h(c(t_0))),h(c(t_1))))\dots,h(c(t_n)))$$

$$\stackrel{ind.\ hyp.}{=} f_{r_n}^{\mathcal{A}}(\dots(f_{r_1}^{\mathcal{A}}(f_{s\to v}^{\mathcal{A}}(fold^{\mathcal{A}}(t_0)),fold^{\mathcal{A}}(t_1)))\dots,fold^{\mathcal{A}}(t_n))$$

$$\stackrel{fold^{\mathcal{A}}\ hom.}{=} fold^{\mathcal{A}}(f_{r_n}(\dots(f_{r_1}(f_{s\to v}(t_0),t_1)\dots),t_n)) \stackrel{(12)}{=} fold^{\mathcal{A}}(t)$$

where for all $1 \leq i \leq n$, $r_i' = (s' \to w_i s')$. ❏

Consequently, when designing a compiler induced by a parser for G' (see above) one may stay with $\Sigma(G)$-algebras as target languages and need not take into account $\Sigma(G')$-algebras:

$$compile_{parse_{G'}} = compile'_{parse_{G'}} =_{def} ((\mathcal{P}(fold_s^{derec(\mathcal{A})}+id_E)\circ parse_{G'}$$
$$: X^* \to \mathcal{P}(\mathcal{A}(s)+E))_{(\mathcal{A},s)\in Alg_{\Sigma(G)}\times S}.$$

9.16 State unfolding

Let $\Sigma = (S, D)$ be a destructive polynomial signature, $C \in Set^S$ and $\mathcal{A}$ be a Σ-algebra with carrier A.

A **coloring of** A **by** C is an S-sorted function $g : A \to C$. From now on, C^A denotes the set of colorings of A by C.

Given $g \in C^A$, the **coextension** $g^\# : A \to DT_\Sigma(C)$ **of** g **to** $DT_\Sigma(C)$, also called **unfolding**, is the $\mathcal{T}_{po}(S)$-sorted function whose values are the labelled trees that are defined as follows:

- For all $I \subseteq \mathcal{I}$, $g_I^\# = id_I$.
- For all $s \in S$ and $a \in A_s$,

$$g_s^\#(a) = g_s(a)\{d \to g_e^\#(d^{\mathcal{A}}(a)) \mid d : s \to e \in D\}. \tag{1}$$

- For all $e = \prod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $a = (a_i)_{i \in I} \in \times_{i \in I} A_{e_i}$,

$$\pi_i(g_e^\#(a)) = g_{e_i}^\#(a_i). \tag{2}$$

- For all $e = \coprod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $a \in A_{e_i}$,

$$g_e^\#(\iota_i(a)) = i(g_{e_i}^\#(a)). \tag{3}$$

This is a mutually inductive definition of the elements of the tuple

$$(g^{\#}(a) : (\mathcal{I} \cup D)^* \to \mathcal{I} \cup C)_{a \in A}$$

of functions.

In particular, (1) is just an abbreviation of two equations:

$$\begin{aligned} g_s^{\#}(a)(\epsilon) &= g_s(a), \\ \forall\, d : s \to e \in D,\ w \in (\mathcal{I} \cup D)^* : g_s^{\#}(a)(dw) &= g_e^{\#}(d^{\mathcal{A}}(a))(w). \end{aligned}$$

Intuitively, $g^{\#}$ unfolds $a \in A$ into a Σ-coterm representing the **behavior** of a in $\mathcal{A}$ and thus computes an **operational semantics** of t [78].

In particular, $id_A^{\#} : \mathcal{A} \to DT_\Sigma(A)$ computes for each $a \in A$ the (tree unfolding of the) transition subgraph of $\mathcal{A}$ with root a.

For all $d : s \to e \in D$ and $a \in A_s$, $id_A^{\#}(a)(d) = d^{\mathcal{A}}(a)$.

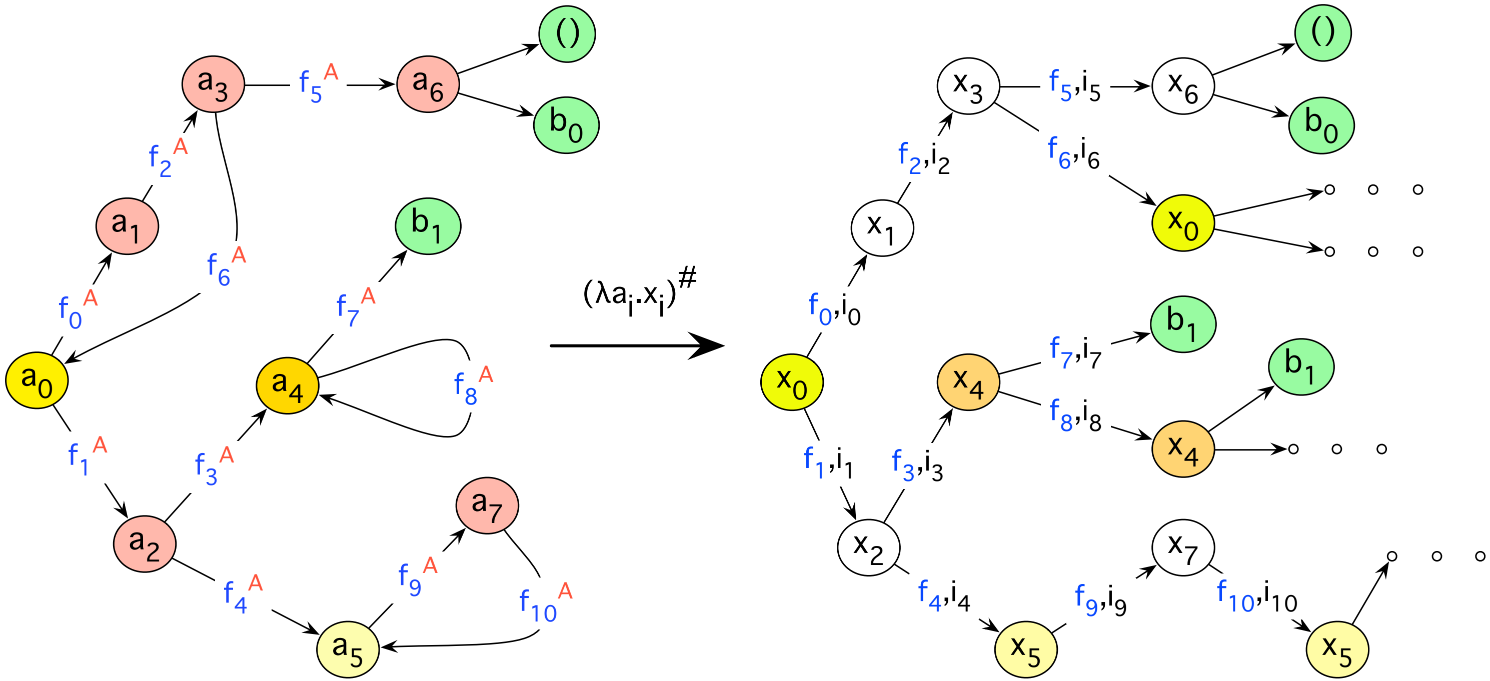


Illustration of state unfolding. $x \xrightarrow{f,i} y$ *stands for* $x \xrightarrow{f} i \xrightarrow{\epsilon} y$.

Given $t \in DT_\Sigma(C)$, $g \in C^A$ **solves** the Σ**-coequation** $ex(t)$ in $\mathcal{A}$, written as $\mathcal{A} \models_g ex(t)$, if $g^{\#}(a) = t$ for some $a \in A$.

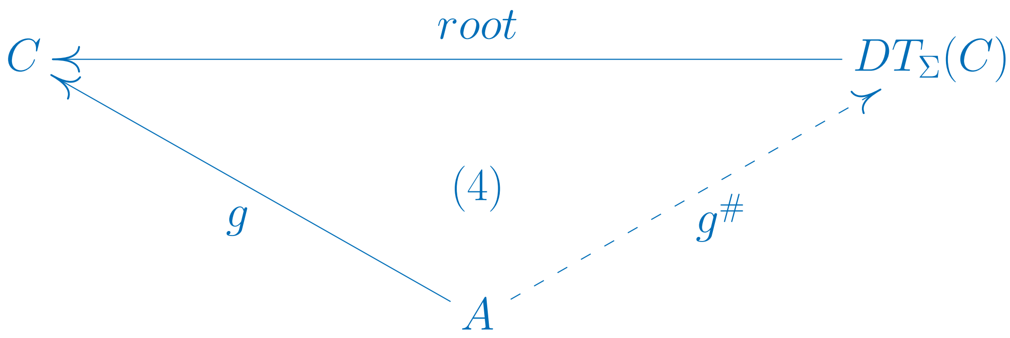
$\mathcal{A}$ **satisfies** a **conditional Σ-coequation** $ex(t) \Rightarrow \bigvee_{i=1}^{n} ex(t_i)$ if every $g \in C^A$ that solves $ex(t)$ also solves $ex(t_i)$ for some $1 \leq i \leq n$.

An empty disjunction is abbreviated to *False*.

Consequently, $\mathcal{A}$ satisfies $\neg ex(t)$ if for all $g \in C^A$ and $a \in A$, $g^\#(a) \neq t$.

Theorem 9.12

Let $C \in Set^S$. $DT_\Sigma(C)$ is a **cofree Σ-algebra over** C, i.e., for all Σ-algebras $\mathcal{A}$ with carrier A and $g \in C^A$, $g^\#$ is the only Σ-homomorphism from $\mathcal{A}$ to $DT_\Sigma(C)$ that satisfies (4).



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_Σ. $g^\#$ no longer depends on g and is denoted by $unfold^{\mathcal{A}}$.

Proof. By (1), $g^\#$ satisfies (4).

$g^\#$ is Σ-homomorphic: For all $a \in A_s$ and $d : s \to e \in D$,

$$d^{DT_\Sigma(C)}(g^\#_s(a)) \stackrel{(1)}{=} d^{DT_\Sigma(C)}(g_s(a)\{d \to g^\#_s(d^{\mathcal{A}}(a)) \mid d : s \to e \in D\}) = g^\#_s(d^{\mathcal{A}}(a)).$$

$g^\#$ is unique: Let $h : \mathcal{A} \to DT_\Sigma(C)$ be a Σ-homomorphism with $root \circ h = g$.

- For all $s \in S$ and $a \in A_s$, $g^\#_s(a)(\epsilon) \stackrel{(1)}{=} g_s(a) \stackrel{root \circ h = g}{=} h_s(a)(\epsilon)$.
- For all $d : s \to e \in D$, $a \in A_s$ and $w \in (\mathcal{I} \cup D)^*$,

$$g^\#_s(a)(dw) \stackrel{(1)}{=} g^\#_e(d^{\mathcal{A}}(a))(w) \stackrel{ind.\ hyp.}{=} h_e(d^{\mathcal{A}}(a))(w) \stackrel{h\ hom.}{=} d^{DT_\Sigma(C)}(h_s(a))(w)$$

$$= h_s(a)(dw).$$

- For all $e = \prod_{i \in I} e_i \in \mathcal{T}_{po}(S)$ and $a \in \prod_{i \in I} A_{e_i}$,

$$\pi_i(g^\#_e(a)) \stackrel{(2)}{=} g^\#_{e_i}(\pi_i(a)) \stackrel{ind.\ hyp.}{=} h_{e_i}(\pi_i(a)) = \pi_i(h_e(a)).$$

- For all $e = \coprod_{i\in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $a \in A_{e_i}$,

$$g_e^\#(\iota_i(a)) \stackrel{(3)}{=} i(g_{e_i}^\#(a)) \stackrel{ind.\ hyp.}{=} i(h_{e_i}(a)) = \iota_i(h_{e_i}(a)) = h_e(\iota_i(a)).$$

Hence for all $e \in \mathcal{T}_{po}(S)$, $g_e^\# = h_e$. ❏

Since $g^\#$ is the only Σ-homomorphism from $\mathcal{A}$ to $DT_\Sigma(C)$ that satisfies (4), $unfold^{\mathcal{A}}$ is the only Σ-homomorphism from $\mathcal{A}$ to DT_Σ, i.e., **DT_Σ is final in Alg_Σ**.

Theorem 9.13

Let $\mathcal{A}$ be Σ-algebra with carrier A, $a \in A$, $w \in (\mathcal{I} \cup D)^*$, $b = id_A^\#(a)(w)$ and $d : s \to e \in D$.

(1) Let $b \in A_s$. Then

$$id_A^\#(a)(wd) = d^{\mathcal{A}}(b).$$

(2) Let $b = \iota_i(c)$ for some $i \in \mathcal{I}$, $c \in A_e$ and $e = \prod_{j\in J} e_{ij} \in \mathcal{T}_s(S)$. Then for all $j \in J$,

$$id_A^\#(a)(wj) = \pi_j(c).$$

(3) $P(a) =_{def} img(id_A^\#(a))$ is the least Σ-invariant $\langle a\rangle$ of $\mathcal{A}$ that contains a (see section 9.9).

Let $\mathcal{B}$ be a Σ-algebra with carrier B and $b \in B$.

(4) For all Σ-homomorphisms $h : \mathcal{B} \to \mathcal{A}$,

$$h(\langle b \rangle) = \langle h(b) \rangle.$$

Let $\mathcal{A}$ be final in Alg_Σ. $(\mathcal{B}, b)$ **realizes** $a \in A$ if $unfold^{\mathcal{B}}(b) = a$. If a is a set, then $(\mathcal{B}, b)$ is also called an **acceptor** of a.

(5) For all $a \in A$, $(\langle a \rangle, a)$ is a minimal realization (acceptor) of a.

Proof of (1) by induction on $|w|$. If $w = \epsilon$, then $b = id_A^\#(a)(w) = id^A(a) = a$ and thus

$$id_A^\#(a)(wd) = id_A^\#(a)(d) = d^{\mathcal{A}}(a) = d^{\mathcal{A}}(b).$$

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

$$id_A^\#(a)(wd) = id_A^\#(x^{\mathcal{A}}(a))(vd) \stackrel{ind.\ hyp.}{=} d^{\mathcal{A}}(id_A^\#(x^{\mathcal{A}}(a))(v)) = d^{\mathcal{A}}(id_A^\#(a)(w)) = d^{\mathcal{A}}(b).$$

Case 2: $a = \iota_i(c) \in A_e$ for some $i \in \mathcal{I}$, $c \in A_e$ and $e = \prod_{j \in J} e_{ij} \in \mathcal{T}_s(S)$. Then $x \in J$ and thus

$$id_A^{\#}(a)(wd) = i(id_A^{\#}(c))(wd) = i(id_A^{\#}(c))(xvd) = id_A^{\#}(\pi_x(c))(vd)$$
$$\overset{ind.\ hyp.}{=} d^{\mathcal{A}}(id_A^{\#}(\pi_x(c)))(v) = i(d^{\mathcal{A}}(id_A^{\#}(c)))(xv) = i(d^{\mathcal{A}}(id_A^{\#}(c)))(w)$$
$$= d^{\mathcal{A}}(i(id_A^{\#}(c))(w)) = d^{\mathcal{A}}(id_A^{\#}(\iota_i(c))(w)) = d^{\mathcal{A}}(id_A^{\#}(a)(w)) = d^{\mathcal{A}}(b).$$

Proof of (2) by induction on $|w|$. If $w = \epsilon$, then $a = id_A(a) = id_A^{\#}(a)(w) = b = \iota_i(c)$ and thus for all $j \in J$,

$$id_A^{\#}(a)(wj) = id_A^{\#}(\iota_i(c))(j) = i(id_A^{\#}(c))(j) = id_A^{\#}(\pi_j(c))(\epsilon) = id_A(\pi_j(c)) = \pi_j(c).$$

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 thus

$$b = id_A^{\#}(a)(w) = id_A^{\#}(a)(xv) = id_A^{\#}(x^{\mathcal{A}}(a))(v).$$

Hence by induction hypothesis, for all $j \in J$, $id_A^{\#}(x^{\mathcal{A}}(a))(vj) = \pi_j(c)$. Therefore,

$$id_A^{\#}(a)(wj) = id_A^{\#}(a)(xvj) = id_A^{\#}(x^{\mathcal{A}}(a))(vj) = \pi_j(c).$$

Case 2: $a = \iota_k(c')$ for some $k \in \mathcal{I}$, $c' \in A_e$ and $e = \prod_{j \in J'} e'_{kj} \in \mathcal{T}_s(S)$. Then $x \in J'$ and thus

$$b = id_A^{\#}(a)(w) = id_A^{\#}(\iota_k(c'))(w) = k(id_A^{\#}(c'))(xv) = id_A^{\#}(\pi_x(c'))(v).$$

Hence by induction hypothesis, for all $j \in J$, $id_A^\#(\pi_x(c'))(vj) = \pi_j(c)$. Therefore,

$$id_A^\#(a)(wj) = id_A^\#(\iota_k(c'))(xvj) = k(id_A^\#(c'))(xvj) = id_A^\#(\pi_x(c'))(vj) = \pi_j(c).$$

Proof of (3).

Since $id_A^\#(a)(\epsilon) = id_A(a) = a$, $P(a)$ contains a.

By (1), for all $s \in S$, $d : s \to e \in D$ and $b \in P(a)_s$, $d^{\mathcal{A}}(b) \in P(a)_e$. Hence $P(a)$ is a Σ-invariant of $\mathcal{A}$.

Let $Q(a)$ be a Σ-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

$$b = id_A^\#(a)(w) = id_A^\#(a)(vx) \stackrel{(1)}{=} x^{\mathcal{A}}(b') \in Q(a)$$

because $b' \in Q(a)$ and $Q(a)$ is a Σ-invariant.

Case 2: $b' = \iota_i(c)$ for some $i \in \mathcal{I}$, $c \in A_e$ and $\prod_{j \in J} e_{ij} \in \mathcal{T}_s(S)$. Then $c \in Q(a)$, $x \in J$ and thus

$$b = id_A^\#(a)(w) = id_A^\#(a)(vx) \stackrel{(2)}{=} \pi_x(c) \in Q(a).$$

Hence $b \in Q(a)$ in both cases, and we conclude that $P(a)$ is the *least* invariant of $\mathcal{A}$ that contains a.

Proof of (4). Since

$$h(\langle b \rangle) = \{h(id_B^\#(b)(w)) \mid w \in (\mathcal{I} \cup D)^*\} \text{ and } \langle h(b) \rangle = \{id_A^\#(h(b))(w) \mid w \in (\mathcal{I} \cup D)^*\},$$

(4) follows from:

$$\forall\ w \in (\mathcal{I} \cup D)^* : h(id_B^\#(b)(w)) = id_B^\#(h(b))(w). \tag{6}$$

Proof of (6) by induction on w. If $w = \epsilon$, then

$$h(id_B^\#(b)(w)) = h(id_B(b)) = h(b) = id_A(h(b)) = id_A^\#(h(b))(w).$$

If $w = vd$ for some $v \in (\mathcal{I} \cup D)^*$ and $d : s \to e \in D$, then

$$h(id_B^\#(b)(wd)) \stackrel{(1)}{=} h(d^{\mathcal{B}}(id_B^\#(b)(w))) \stackrel{h\ hom.}{=} d^{\mathcal{A}}(h(id_B^\#(b)(w))) \stackrel{ind.\ hyp.}{=} d^{\mathcal{A}}(id_A^\#(h(b))(w))$$
$$\stackrel{(1)}{=} id_A^\#(h(b))(wd).$$

If $id_B^\#(b)(w) = \iota_i(c)$ and $w = vj$ for some $i \in \mathcal{I}$, $c \in B_e$, $\prod_{j \in J} e_{ij} \in \mathcal{T}_s(S)$ and $j \in J$, then

$$h(id_B^\#(b)(w)) \overset{h\ hom.}{=} \iota_i(h(c)) \tag{7}$$

and thus $h(id_B^\#(b)(wj)) \overset{(2)}{=} h(\pi_j(c)) \overset{h\ hom.}{=} \pi_j(h(c)) \overset{(2),(7)}{=} id_A^\#(h(b))(wj)$.

This ends the proofs of (6) and thus of (4).

Proof of (5). Since $\mathcal{A}$ is final in Alg_Σ, $unfold^\mathcal{A} = id_A$. Hence for all $a \in A$,

$$a = unfold^\mathcal{A}(a) = unfold^\mathcal{A}(inc_{\langle a\rangle}(a)) = unfold^{\langle a\rangle}(a)$$

and thus $(\langle a\rangle, a)$ realizes t. Moreover, let $(\mathcal{B}, b)$ realize a. Then

$$|\langle a\rangle| = |\langle unfold^\mathcal{A}(b)\rangle| \overset{(4)}{=} |unfold^\mathcal{A}(\langle b\rangle)| \leq |\langle b\rangle|,$$

and thus $(\langle a\rangle, a)$ is minimal. ❏

If $\mathcal{A} = DT_\Sigma(C)$, then $g \in C^A$ is called a **recoloring** because in this case $g^\# : \mathcal{A} \to DT_\Sigma(C)$ modifies a coterm t by simply changing the node labels of t: For all subcoterms u of t, $g^\#(t)$ labels the root of u with $g(u)$.

Corollary 9.14 For all $t \in DT_\Sigma(C)$, $\langle t\rangle$ is the set of subtrees of t, i.e., for all $v \in (\mathcal{I} \cup D)^*$,

$$\lambda w.t(vw) = h(v) =_{def} id^\#_{DT_\Sigma(C)}(t)(v).$$

Proof by induction on $|v|$. If $v = \epsilon$, then

$$\lambda w.t(vw) = \lambda w.t(w) = t = id_{DT_\Sigma(C)}(t) = id^\#_{DT_\Sigma(C)}(t)(\epsilon) = h(\epsilon) = h(v).$$

Otherwise $v = v'x$ for some $v' \in (\mathcal{I} \cup D)^*$ and $x \in \mathcal{I} \cup D$. Hence

$$\lambda w.t(vw) = \lambda w.t(v'xw) = \lambda w.(\lambda w'.t(v'w'))(xw) \overset{ind.\ hyp.}{=} \lambda w.h(v')(xw). \tag{6}$$

Case 1. $h(v') \in DT_\Sigma(C)_s$ for some $s \in S$. Then $x \in D$ and thus

$$\lambda w.t(vw) \overset{(6)}{=} \lambda w.h(v')(xw) = \lambda w.x^{DT_\Sigma(C)}(h(v'))(w) = x^{DT_\Sigma(C)}(h(v')) \overset{(1)}{=} h(vx) = h(v).$$

Case 2. $h(v') \in DT_\Sigma(C)_e$ for some $e = \coprod_{i\in I} e_i \in \mathcal{T}_{po}(S)$. 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

$$\lambda w.t(vw) \stackrel{(6)}{=} \lambda w.h(v')(xw) = \lambda w.i(u)(()w) = \lambda w.u(w) = u = h(v'()) = h(v'x) = h(v).$$

Case 3. $h(v') \in DT_\Sigma(C)_e$ for some $e = \prod_{i\in I} e_i \in \mathcal{T}_{po}(S)$. Then $x = i$ for some $i \in I$ and thus

$$\lambda w.t(vw) \stackrel{(6)}{=} \lambda w.h(v')(xw) = \lambda w.\pi_i(h(v'))(w) = \pi_i(h(v')) \stackrel{(2)}{=} h(v'i) = h(v'x) = h(v). \quad ❑$$

A Σ-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_Σ whose objects are all observable Σ-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 12.1 (2), for all $\mathcal{A} \in OAlg_\Sigma$, $\mathcal{A} \cong DT_\Sigma|_{img(unfold^{\mathcal{A}})}$.

Lemma 9.15

Let $\mathcal{K}$ be a full subcategory of $OAlg_\Sigma$ and the Σ-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 equation 2.5.20) and $\mathcal{B}(s) = DT_{\Sigma,s}|_{B_s}$.
- For all $d : s \to e \in D$ and $t \in DT_\Sigma$, $d^{\mathcal{B}}(t) = d^{DT_\Sigma}(t)$.

For all $\mathcal{A} \in \mathcal{K}$, there is a unique Σ-homomorphism from $\mathcal{A}$ to $\mathcal{B}$.
In particular, $\mathcal{B}$ is final in $OAlg_\Sigma$.

Proof. By Lemma 12.1 (2), there is a unique Σ-epimorphism $h : \coprod \mathcal{K} \to \mathcal{B}$ such that $inc_B \circ h = [unfold^{\mathcal{A}}]_{\mathcal{A}\in\mathcal{K}}$. Hence for all $\mathcal{A} \in \mathcal{K}$, $h \circ \iota_{\mathcal{A}} : \mathcal{A} \to \mathcal{B}$ is Σ-homomorphic. Suppose that there are two Σ-homomorphisms $h_1, h_2 : \mathcal{A} \to \mathcal{B}$. Since DT_Σ is final in Alg_Σ, $inc_B \circ h_1 = inc_B \circ h_2$. Hence $h_1 = h_2$ because inc_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 Σ-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 [161], section 5, and embedded in the final observable Σ-algebra. ❏

Lemma 9.16

For all Σ-algebras $\mathcal{A}$ with carrier A, $g \in C^A$ and Σ-homomorphisms $h : \mathcal{B} \to \mathcal{A}$,

$$(g \circ h)^\# = g^\# \circ h.$$

Proof. Since $root \circ g^\# \circ h = g \circ h$, the conjecture follows from the fact that $(g \circ h)^\#$ is the only Σ-homomorphism $h' : \mathcal{B} \to DT_\Sigma(C)$ with $root \circ h' = g \circ h$. ❏

9.17 Coterm grounding

Let $C \in \mathcal{I}$.

$$\Sigma(C) = (S, D \cup \{col_s : s \to C_s \mid s \in S\})$$

is called the **grounding of Σ 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 $h, h' : \mathcal{A} \to DT_\Sigma(C)$ 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 Σ-homomorphic, we conclude $h = h'$.

Therefore, **$DT_\Sigma(C)$ 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 ϵ 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 Σ-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 chapter 15).

Final models of weighted types M_C^e (see chapter 7) are quotients of polynomial weighted types, i.e., types of the form $(M \times e)_C^*$ (see chapter 15).

9.18 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 9.6.2), (1)
- $DT_{Stream(X)} \cong InfSeq(X)$ is final in Alg_{Stream} (see sample algebra 9.6.5), (2)
- $DT_{coDyn(X,Y)} \cong coSeq(X,Y)$ is final in $Alg_{coDyn(X,Y)}$ (see sample algebra 9.6.8), (3)
- $DT_{coNelist(X)} \cong Neseq(X)$ is final in $Alg_{coNelist(X)}$ (see sample algebra 9.6.9), (4)
- $DT_{infBintree(X)} \cong InfBin(X)$ is final in $Alg_{infBintree(X)}$ (see sample algebra 9.6.11), (5)
- $DT_{coBintree(X)} \cong Bin(X)$ is final in $Alg_{coBintree(X)}$ (see sample algebra 9.6.12), (6)
- $DT_{infTree(X)} \cong FBInfTree(X)$ is final in $Alg_{infTree(X)}$ (see sample algebra 9.6.16), (7)
- $DT_{coTree_\omega(X)} \cong FBTree(X)$ is final in $Alg_{coTree_\omega(X)}$ (see sample algebra 9.6.17), (8)
- $DT_{coTree(X)} \cong Tree_\infty(X)$ is final in $Alg_{coTree(X)}$ (see sample algebra 9.6.18), (9)
- $DT_{Acc(X)} \cong Pow(X)$ is final in $Alg_{Acc(X)}$ (see sample algebra 9.6.20), (10)
- $DT_{NAcc(X)} \cong NPow(X)$ is final in $Alg_{NAcc(X)}$ (see sample algebra 9.6.21), (11)
- $DT_{DAut(X,Y)} \cong Beh(X,Y)$ is final in $Alg_{DAut(X,Y)}$ (see sample algebra 9.6.24), (12)
- $DT_{Mealy(X,Y)} \cong MBeh(X,Y) \cong Causal(X,Y)$ is final in $Alg_{Mealy(X,Y)}$ (see sample algebra 9.6.26), (13)
- $DT_{PAut(X,Y)} \cong PBeh(X,Y)$ is final in $Alg_{PAut(X,Y)}$ (see sample algebra 9.6.27), (14)

- $DT_{NAut^*(X,Y)} \cong NBeh(X,Y)$ is final in $Alg_{NAut^*(X,Y)}$ (see sample algebra 9.6.28),(15)
- $DT_{TAcc(\Sigma)} \cong TPow(\Sigma)$ is final in $Alg_{TAcc(\Sigma)}$ (see sample algebra 9.6.29), (16)
- $DT_{NTAcc(\Sigma)} \cong NTPow(\Sigma)$ is final in $Alg_{NTAcc(\Sigma)}$ (see sample algebra 9.6.30), (17)
- $DT_{NTAcc^*(\Sigma)} \cong NTPow^*(\Sigma)$ is final in $Alg_{NTAcc^*(\Sigma)}$ (see sample algebra 9.6.31), (18)
- $DT_{Med(X)} \cong 1$ is final in $Alg_{Med(X)}$,
- $DT_{NMed^*(X)} \cong NPow^*(X)$ is final in $Alg_{NMed^*(X)}$ (see sample algebra 9.6.32). (19)

Moreover, by (10) 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 19.5).

Given a destructive signature Σ and a Σ-algebra $\mathcal{A}$ with carrier A, $unfold^{\mathcal{A}}$ denotes the unique Σ-homomorphism from $\mathcal{A}$ not only to DT_Σ, but also to isomorphic representations of DT_Σ like those listed above.

In cases (1)-(18), the respective definition of $h =_{def} unfold^{\mathcal{A}}$ reads as follows:

1. *coNat*-algebra $\mathbb{N}_\infty$

$$\begin{aligned} h : A &\to \mathbb{N}_\infty \\ a &\mapsto max\{n \in \mathbb{N}_\infty \mid \forall\, 0 \leq i < n : (pred^{\mathcal{A}} + id_1)^i(pred^{\mathcal{A}}(a)) \neq ()\} \end{aligned}$$

where max denotes the maximum w.r.t. the usual (well-founded) ordering $<$ on $\mathbb{N}_\infty$ - with $n < \omega$ for all $n \in \mathbb{N}$ (see, e.g., [6], Examples 3.8) .

h is *coNat*-homomorphic:

Case 1: $pred^{\mathcal{A}}(a) = ()$. Then $h(a) = 0$. Hence

$$pred^{\mathbb{N}_\infty}(h(a)) = pred^{\mathbb{N}_\infty}(0) = () = pred^{\mathcal{A}}(a).$$

Case 2: $h(pred^{\mathcal{A}}(a)) \in \mathbb{N}$. Then $h(a) = h(pred^{\mathcal{A}}(a)) + 1$. Hence

$$pred^{\mathbb{N}_\infty}(h(a)) = pred^{\mathbb{N}_\infty}(h(pred^{\mathcal{A}}(a)) + 1) = h(pred^{\mathcal{A}}(a)).$$

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 $h' : \mathcal{A} \to \mathbb{N}_\infty$ be *coNat*-homomorphic.

Case 1: $h'(a) = 0$. Then

$$\epsilon = pred^{\mathbb{N}_\infty}(h'(a)) \overset{h'\ hom.}{=} pred^{\mathcal{A}}(a)$$

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

$$h(pred^{\mathcal{A}}(a)) \overset{ind.\ hyp.}{=} h'(pred^{\mathcal{A}}(a)) \overset{h'\ hom.}{=} pred^{\mathbb{N}_\infty}(h'(a)) = h'(a) - 1$$

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 $a \gg b \Leftrightarrow_{def} h(a) > h(b)$. 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

$$\infty = pred^{\mathbb{N}_\infty}(\infty) = pred^{\mathbb{N}_\infty}(h'(a))) \overset{h'\ hom.}{=} h'(pred^{\mathcal{A}}(a)) \overset{ind.\ hyp.}{=} h(pred^{\mathcal{A}}(a)).$$

Hence $h(a) = \omega = h'(a)$.

Exercise 18 Define a Σ-algebra $\mathcal{A}$ with carrier $A = \mathbb{N}_\infty^2$ such that

$$\begin{aligned} +_\infty : \mathbb{N}_\infty^2 &\to \mathbb{N}_\infty \\ (m,n) &\mapsto \mathit{if}\ m,n \in \mathbb{N}\ \mathit{then}\ m+n\ \mathit{else}\ \infty \end{aligned}$$

agrees with $unfold^{\mathcal{A}}$. ❏

Analogously to the definition of unfolding into coterms (see section 9.16), the following definitions of $h : A \to (X_1 \to \cdots \to (X_n \to X)\ldots)$ are mutually inductive definitions of the elements of the tuple

$$(h(a)(x_1)\ldots(x_n))_{a\in A, x_1\in X_1,\ldots,x_n\in X_n}.$$

2. *Stream*(X)-algebra *InfSeq*(X)

$$\begin{aligned} h : A &\to X^{\mathbb{N}} \\ a &\mapsto \lambda n.\mathit{if}\ n = 0\ \mathit{then}\ head^{\mathcal{A}}(a)\ \mathit{else}\ h(tail^{\mathcal{A}}(a))(n-1) \end{aligned}$$

3. $coDyn(X,Y)$-algebra $coSeq(X,Y)$

$$
\begin{array}{rcl}
h : A & \to & X^* \times Y \cup X^{\mathbb{N}} \\
a & \mapsto & \begin{cases}
(xw, y) & \text{if } split^{\mathcal{A}}(a) = \iota_1(x,b) \wedge h(b) = (w,y) \in X^* \times Y \\
(\epsilon, y) & \text{if } split^{\mathcal{A}}(a) = \iota_2(y) \\
\lambda n.if\ \ n = 0\ then\ x & \\
\qquad else\ h(b)(n-1) & \text{if } split^{\mathcal{A}}(a) = \iota_1(x,b) \wedge h(b) \in X^{\mathbb{N}}
\end{cases}
\end{array}
$$

4. $coNelist(X)$-algebra $Neseq(X)$

$$
\begin{array}{rcl}
h : A & \to & X^{+} \cup X^{\mathbb{N}} \\
a & \mapsto & \begin{cases}
xw & \text{if } split^{\mathcal{A}}(a) = \iota_1(x,b) \wedge h(b) = w \in X^* \\
\epsilon & \text{if } split^{\mathcal{A}}(a) = \iota_2() \\
\lambda n.if\ \ n = 0\ then\ x & \\
\qquad else\ h(b)(n-1) & \text{if } split^{\mathcal{A}}(a) = \iota_1(x,b) \wedge h(b) \in X^{\mathbb{N}}
\end{cases}
\end{array}
$$

5. *infBintree*(X)-algebra *InfBin*(X)

$$\begin{aligned} h : A &\rightarrow X^{2^*} \\ a &\mapsto \lambda w.if\ w = \epsilon\ then\ root^{\mathcal{A}}(a) \\ &\qquad else\ if\ head(w) = 0\ then\ h(left^{\mathcal{A}}(a))(tail(w))\ else\ h(right^{\mathcal{A}}(a))(tail(w)) \end{aligned}$$

6. *coBintree*(X)-algebra *Bin*(X)

$$\begin{aligned} h : A &\rightarrow ltr(2, X) \\ a &\mapsto \begin{cases} x\{0 \rightarrow h(b), 1 \rightarrow h(c)\} & \text{if } split^{\mathcal{A}}(a) = \iota_1(x, b, c) \\ \Omega & \text{if } split^{\mathcal{A}}(a) = \iota_2() \end{cases} \end{aligned}$$

7. *infTree*(X)-algebra *FBInfTree*(X)

$$\begin{aligned} h : A &\rightarrow otr(\mathbb{N}, X) \cap fbtr(\mathbb{N}, X) \cap itr(\mathbb{N}, X) \\ a &\mapsto root^{\mathcal{A}}(a)(h(a_1), \ldots, h(a_n)) \text{ where } (a_1, \ldots, a_n) = subtrees^{\mathcal{A}}(a) \end{aligned}$$

8. *coTree*$_\omega(X)$-algebra *FBTree*(X)

$$\begin{aligned} h : A &\rightarrow otr(\mathbb{N}, X) \cap fbtr(\mathbb{N}, X) \\ a &\mapsto \begin{cases} root^{\mathcal{A}}(a)(h(a_1), \ldots, h(a_n)) & \text{if } subtrees^{\mathcal{A}}(a) = (a_1, \ldots, a_n) \\ root^{\mathcal{A}}(a) & \text{if } subtrees^{\mathcal{A}}(a) = \epsilon \end{cases} \end{aligned}$$

9. $coTree(X)$-algebra $Tree_\infty(X)$

$$
\begin{array}{rcl}
h_{tree} : A_{tree} & \rightarrow & otr(\mathbb{N}, X) \\
a & \mapsto & \begin{cases} root^{\mathcal{A}}(a)(t_1, \dots, t_n) & \text{if } h_{trees}(subtrees^{\mathcal{A}}(a)) = (t_1, \dots, t_n) \\ root^{\mathcal{A}}(a) & \text{if } h_{trees}(subtrees^{\mathcal{A}}(a)) = \epsilon \\ root^{\mathcal{A}}(a)\{n \rightarrow t_n \mid n \in \mathbb{N}\} & \text{if } h_{trees}(subtrees^{\mathcal{A}}(a)) = (t_n)_{n \in \mathbb{N}} \end{cases} \\
h_{trees} : A_{trees} & \rightarrow & otr(\mathbb{N}, X)^\infty \\
as & \mapsto & \begin{cases} h_{tree}(a) \cdot h_{trees}(bs) & \text{if } split^{\mathcal{A}}(as) = \iota_1(a, bs) \\ \epsilon & \text{if } split^{\mathcal{A}}(as) = \iota_2() \end{cases}
\end{array}
$$

10. $Acc(X)$-algebra $Pow(X)$

$$
\begin{array}{rcl}
h : A & \rightarrow & \mathcal{P}(X^*) \\
a & \mapsto & \begin{cases} \{x \cdot w \mid x \in X,\ w \in h(\delta^{\mathcal{A}}(a)(x))\} & \text{if } \beta^{\mathcal{A}}(a) = 0 \\ \{x \cdot w \mid x \in X,\ w \in h(\delta^{\mathcal{A}}(a)(x))\} \cup 1 & \text{if } \beta^{\mathcal{A}}(a) = 1 \end{cases}
\end{array}
$$

For all $a \in A$, $(\mathcal{A}, a)$ **accepts** the **language** $h(a)$.

Given a language $L \subseteq X^*$, Theorem 9.13 (5) implies that $(\langle L \rangle, L)$ is a minimal acceptor of L. The final states of $(\langle L \rangle, L)$ are the languages of $\langle L \rangle$ that contain ϵ.

11. $NAcc(X)$-algebra $NPow(X)$

$$
\begin{array}{rcl}
h : A & \to & \mathcal{P}(X^*) \\
a & \mapsto & \begin{cases} \{x \cdot w \mid x \in X,\ \exists\, b \in \delta^{\mathcal{A}}(a)(x) : w \in h(b)\} & \text{if } \beta^{\mathcal{A}}(a) = 0 \\ \{x \cdot w \mid x \in X,\ \exists\, b \in \delta^{\mathcal{A}}(a)(x) : w \in h(b)\} \cup 1 & \text{if } \beta^{\mathcal{A}}(a) = 1 \end{cases}
\end{array}
$$

For all $a \in A$, $(\mathcal{A}, a)$ **accepts** the **language** $h(a)$.

Given a language $L \subseteq X^*$, Theorem 9.13 (5) implies that $(\langle L \rangle, L)$ is a minimal acceptor of L. The final states of $(\langle L \rangle, L)$ are the languages of $\langle L \rangle$ that contain ϵ.

12. $DAut(X, Y)$-algebra $Beh(X, Y)$

$$
\begin{array}{rcl}
h : A & \to & Y^{X^*} \\
a & \mapsto & \lambda w.\mathit{if}\ w = \epsilon\ \mathit{then}\ \beta^{\mathcal{A}}(a)\ \mathit{else}\ h(\delta^{\mathcal{A}}(a)(\mathit{head}(w)))(\mathit{tail}(w))
\end{array}
$$

For all $a \in A$, $(\mathcal{A}, a)$ **realizes** the **behavior function** $h(a)$.

Given a behavior function $f : X^* \to Y$, Theorem 9.13 (5) implies that $(\langle f \rangle, f)$ is a minimal realization of f.

Let Y be a semiring. Then $Beh(X,Y)$ is a linear automaton (see sample algebra 9.6.25). Moreover, if A is a Y-semimodule, then h is linear (see [33], Theorem 2). Consequently, $Beh(X,Y)$ is final in the category $Alg_\Sigma \cap SMod_R$.

13. $Mealy(X,Y)$-algebra $MBeh(X,Y)$

$$\begin{aligned} h : A &\to Y^{X^+} \\ a &\mapsto \lambda w.if\ |w| = 1\ then\ \beta^{\mathcal{A}}(a)(head(w))\ else\ h(\delta^{\mathcal{A}}(a)(head(w)))(tail(w)) \end{aligned}$$

$Mealy(X,Y)$-algebra $Causal(X,Y)$ (see chapter 2):

$$\begin{aligned} h : A &\to \mathcal{C}(X,Y) \\ a &\mapsto \lambda f \lambda n.if\ n = 0\ then\ \beta^{\mathcal{A}}(a)(f(0)) \\ &\qquad\qquad\qquad\quad else\ h(\delta^{\mathcal{A}}(a)(f(0)))(\lambda k.f(k+1))(n-1) \end{aligned}$$

14. $PAut(X,Y)$-algebra $PBeh(X,Y)$

$$\begin{aligned} h : A &\to ltr(X,Y) \\ a &\mapsto \beta^{\mathcal{A}}(a)\{x \to h(b) \mid x \in X,\ \delta^{\mathcal{A}}(a)(x) = b \in A\} \end{aligned}$$

15. $NAut^*(X,Y)$-algebra $NBeh(X,Y)$

$$\begin{array}{rcl} h : A & \to & otr(X \times \mathbb{N}, Y) \\ a & \mapsto & \beta^{\mathcal{A}}(a)\{(x,i) \to h(a_i) \mid x \in X,\ 1 \leq i \leq n,\ \delta^{\mathcal{A}}(a)(x) = (a_1, \ldots, a_n)\} \end{array}$$

Exercise 19 Show that h is Σ-homomorphic, i.e., for all $a \in A_{state}$,

$$\begin{array}{rcl} \delta^{NBeh(X,Y)}(h(a)) & = & map(h) \circ \delta^{\mathcal{A}}(a), \\ \beta^{NBeh(X,Y)}(h(a)) & = & \beta^{\mathcal{A}}(a). \end{array}$$

Let $\Sigma = (S, C)$ be a finitary signature.

16. $TAcc(\Sigma)$-algebra $TPow(\Sigma)$

$$\begin{array}{rcl} h : A & \to & \mathcal{P}(T_\Sigma) \\ a & \mapsto & \{c(t_1, \ldots, t_n) \mid c \in C,\ \delta_c^{\mathcal{A}}(a) = (a_1, \ldots, a_n), \\ & & \qquad \forall\, 1 \leq i \leq n : t_i \in h(a_i)\} \end{array}$$

17. $NTAcc(\Sigma)$-algebra $NTPow(\Sigma)$

$$\begin{array}{rcl} h : A & \to & \mathcal{P}(T_\Sigma) \\ a & \mapsto & \{c(t_1, \ldots, t_n) \mid c \in C,\ \delta_c^{\mathcal{A}}(a) = \{(a_{ij})_{j=1}^n \mid 1 \leq i \leq k\} \\ & & \qquad \Rightarrow\ \forall\, 1 \leq j \leq n\ \exists\, 1 \leq i_j \leq k : t_i \in h(a_{ij})\} \end{array}$$

18. $NTAcc^*(\Sigma)$-algebra $NTPow^*(\Sigma)$

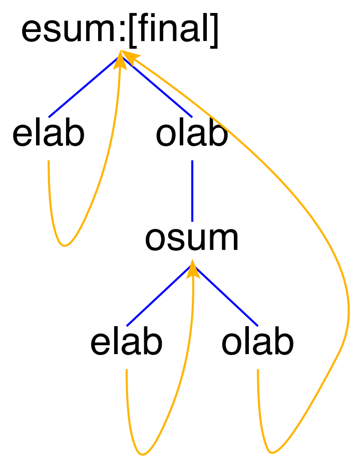
$$
\begin{aligned}
h : A &\to \mathcal{P}(T_\Sigma) \\
a &\mapsto \{c(t_1,\dots,t_n) \mid c \in C,\ \delta_c^{\mathcal{A}}(a) = [(a_{ij})_{j=1}^n \mid 1 \le i \le k] \\
&\qquad \Rightarrow \forall\, 1 \le j \le n\ \exists\, 1 \le i_j \le k : t_i \in h(a_{ij})\}
\end{aligned}
$$

Example 9.17 (see sample algebras 9.6.7 and 9.6.20)

Define $h : eo \to Pow(\mathbb{Z})$ as follows:

$$
\begin{aligned}
h(esum) &= \{(x_1,\dots,x_n) \in \mathbb{Z}^* \mid \textstyle\sum_{i=1}^n x_i \text{ is even}\}, \\
h(osum) &= \{(x_1,\dots,x_n) \in \mathbb{Z}^* \mid \textstyle\sum_{i=1}^n x_i \text{ is odd}\}.
\end{aligned}
$$

Transitions and atoms of eo:



$$(eo, esum) \text{ accepts } h(esum). \tag{1}$$

$$(eo, osum) \text{ accepts } h(osum). \tag{2}$$

Proof of (1) and (2). By Lemma 13.3 (2) and (3), all $DAut(\mathbb{Z}, 2)$-homomorphisms from eo to $Pow(\mathbb{Z})$ agree with $unfold^{eo} : eo \to Pow(\mathbb{Z})$.

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 $x \in \mathbb{Z}$,

$$h(\delta^{eo}(st)(x)) = \delta^{Pow}(h(st))(x), \tag{3}$$
$$\beta^{eo}(st) = \beta^{Pow}(h(st)). \tag{4}$$

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

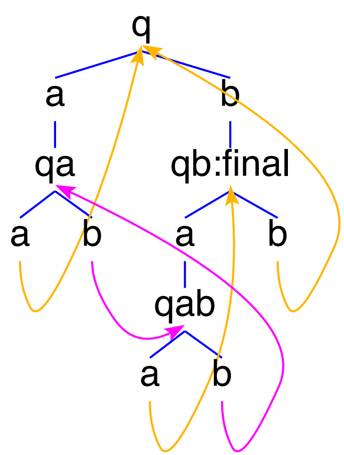
$$w \in h(\delta^{eo}(osum)(x)) \Leftrightarrow w \in h(esum) \Leftrightarrow x \cdot w \in h(osum) \Leftrightarrow w \in \delta^{Pow}(h(osum))(x).$$

Proof of (4). Since $\epsilon \in h(esum)$, $\beta^{eo}(esum) = 1 = \beta^{Pow}(h(esum))$. Since $\epsilon \notin h(osum)$, $\beta^{eo}(osum) = 0 = \beta^{Pow}(h(osum))$. ❏

Exercise 20 Let $\mathcal{A}$ be the following $DAut(\{a, b\}, 2)$-algebra:

$$\begin{aligned} \mathcal{A}_{state} &= \{q, q_a, q_b, q_{ab}\}, \\ \beta^{\mathcal{A}} &= \lambda st.if\ st = qb\ then\ 1\ else\ 0, \end{aligned}$$

$$\delta^{\mathcal{A}}(q)(a) = q_a, \quad \delta^{\mathcal{A}}(q)(b) = q_b, \quad \delta^{\mathcal{A}}(q_a)(a) = q, \quad \delta^{\mathcal{A}}(q_a)(b) = q_{ab},$$
$$\delta^{\mathcal{A}}(q_b)(a) = q_{ab}, \quad \delta^{\mathcal{A}}(q_b)(b) = q, \quad \delta^{\mathcal{A}}(q_{ab})(a) = q_b, \quad \delta^{\mathcal{A}}(q_{ab})(b) = q_a.$$



Define $h : \mathcal{A} \to Pow(\{a,b\})$ as follows

$$\begin{array}{rcl} h(q) & = & \{w \in \{a,b\}^* \mid \exists\, i,j \in \mathbb{N} : \#a(w) = 2*i \wedge \#b(w) = 2*j+1\}, \\ h(q_a) & = & \{w \in \{a,b\}^* \mid \exists\, i,j \in \mathbb{N} : \#a(w) = 2*i+1 \wedge \#b(w) = 2*j+1\}, \\ h(q_b) & = & \{w \in \{a,b\}^* \mid \exists\, i,j \in \mathbb{N} : \#a(w) = 2*i \wedge \#b(w) = 2*j\}, \\ h(q_{ab}) & = & \{w \in \{a,b\}^* \mid \exists\, i,j \in \mathbb{N} : \#a(w) = 2*i+1 \wedge \#b(w) = 2*j\}. \end{array}$$

Prove that $(\mathcal{A}, q)$ accepts $h(q)$ by showing that h is $DAut(\{a,b\},2)$-homomorphic—analogously to Example 9.17. ❏

By Theorem 16.5, for all $t \in T_{Reg(X),state}$, $\langle t \rangle$ is finite. **** 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 [165], section 4).

This fact allows us to build generic top-down parsers for all regular languages over X and to extend them to parsers for context-free languages by simply incorporating the respective grammar rules (see sample biinductive definitions 16.5.6 and 16.5.7 or [136], chapters 15 and 16).

9.19 Σ-flowcharts

While terms denote both objects and computations, coterms denote only objects insofar as—intuitively—the behaviour a coterm 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?

Flowcharts often come as graphs with cycles repesenting iterative and thus possibly infinite computations. Their use for describing iterative control structures is a topic of chapter 17.

Flowcharts with cycles can be unfolded to infinite trees. Hence we define them like terms, but with product and sum types and their respective ingredients exchanged:

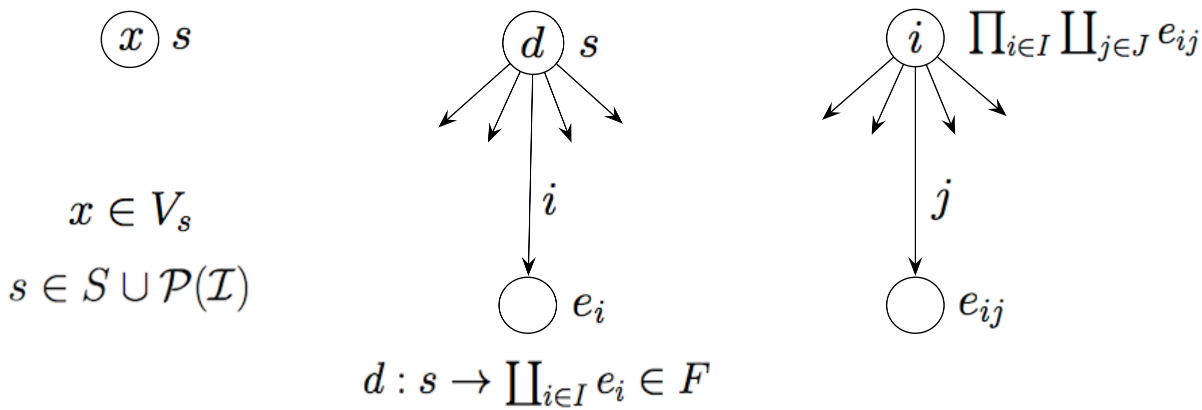
Let $\Sigma = (S, D)$ be a destructive polynomial signature and V be an S-sorted set of "variables".

The set $\overline{CT_\Sigma}(V)$ of **Σ-flowcharts over V** is the greatest $\mathcal{T}_p(S)$-sorted set M 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 $d : s \to \coprod_{i \in I} e_i \in D$ and $u \in \times_{i \in I} M_{e_i}$ such that $t = d(u)$, (1)
- for all $e = \coprod_{i \in I} \prod_{j \in J} e_{ij} \in \mathcal{T}_{po}(S)$ and $t \in M_e$ there are $i \in I$ and $u \in \times_{j \in J} M_{e_{ij}}$ such that $t = i(u)$. (2)

The subset $\overline{T_\Sigma}(V)$ of $\overline{CT_\Sigma}(V)$ of **well-founded Σ-flowcharts over** V is the least $\mathcal{T}_p(S)$-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 $s \in S$, $V_s \subseteq M_s$, (3)
- for all $d : s \to \coprod_{i \in I} e_i \in D$ and $t \in \times_{i \in I} M_{e_i}$, $d(t) \in M_s$, (4)
- for all $e = \prod_{i \in I} \coprod_{j \in J} e_{ij} \in \mathcal{T}_{po}(S)$, $i \in I$ and $t \in \times_{j \in J} M_{e_{ij}}$, $i(t) \in M_e$. (5)



(1/3) (1/4) (2/5)

Intuitively, Σ-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 Σ-terms insofar as sums and products are exchanged here.

Remember that every leaf of a Σ-term is labelled with an element of $\mathcal{I}$ or a variable (regarded as an *entrance* to the term).

In contrast to that, *all* leaves of a Σ-flowchart are labelled with variables and regarded as *exits* from the flowchart.

Let O be a set of "output variables". Given a **flowchart substitution**, i.e., an S-sorted function $g : V \to \overline{T_\Sigma}(O)$, the **flowchart instantiation** $g^* : \overline{T_\Sigma}(V) \to \overline{T_\Sigma}(O)$ is the $\mathcal{T}_{po}(S)$-sorted function that is defined inductively as follows:

- For all $I \subseteq \mathcal{I}$, $g^*_I = id_I$.
- For all $s \in S$ and $x \in V_s$, $g^*_s(x) = g_s(x)$.
- For all $d : s \to e \in D$ and $t \in \overline{T_\Sigma}(V)_e$,

$$g^*_e(d(t)) = d(g^*_s(t)). \tag{1}$$

- For all $e = (e_i)_{i \in I} \in \mathcal{T}_{po}(S)^I$ and $t = (t_i)_{i \in I} \in \mathsf{X}_{i \in I} \overline{T_\Sigma}(V)_{e_i}$,

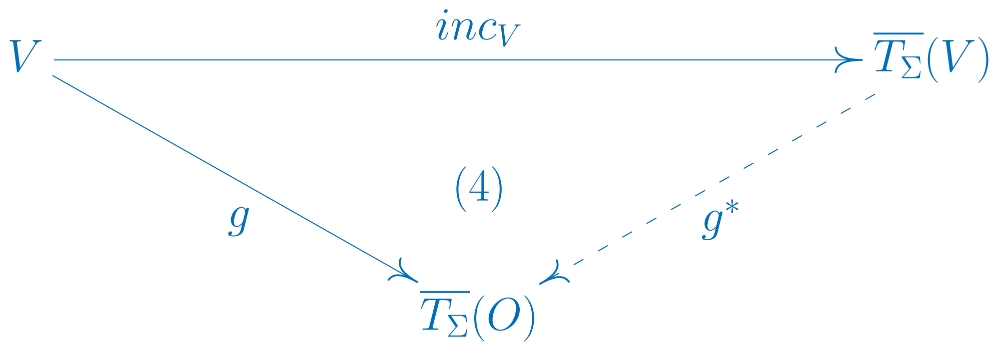
$$g_e^*(t) = (g_{e_i}^*(t_i))_{i \in I}. \tag{2}$$

- For all $e = (e_i)_{i \in I} \in \mathcal{T}_{po}(S)^I$, $i \in I$ and $t \in \overline{T_\Sigma}(V)_{e_i}$,

$$g_e^*(i(t)) = i(g_{e_i}^*(t)). \tag{3}$$

Theorem 9.18

Let $V \in Set^S$. Given an S-sorted function $g : V \to \overline{T_\Sigma}(O)$, g^* is the only $\mathcal{T}_{po}(S)$-sorted function from $\overline{T_\Sigma}(V)$ to $\overline{T_\Sigma}(O)$ that satisfies (1)-(4).



Proof. Analogously to Theorem 9.7. ❑

Let $\mathcal{A}$ be a Σ-algebra with carrier A, $V \times A =_{def} (V_s \times A_s)_{s \in S}$ and A_V be the $\mathcal{T}_{po}(S)$-sorted set that is defined as follows:

- For all $I \subseteq \mathcal{I}$, $A_I = I$.
- For all $s \in S$, $A_{V,s} = A_s \to V \times A$.
- For all $e = \coprod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $A_{V,e} = \coprod_{i \in I} A_{e_i} \to V \times A$.
- For all $e = \prod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $A_{V,e} = \prod_{i \in I} A_{e_i} \to V \times A$.

An S-sorted function from V to A_O is called a **flowchart valuation of V in A_O**.

From now on, A_O^V denotes the set of flowchart valuations of V in A_O.

Given $g \in A_O^V$, the **flowchart extension** $g^\circ : \overline{T_\Sigma}(V) \to A_O^V$, also called **flowchart traversal**, is the $\mathcal{T}_{po}(S)$-sorted function that is defined inductively as follows:

- For all $I \subseteq \mathcal{I}$, $g_I^\circ = id_I$.
- For all $x : s \in V$ and $a \in A_s$, $g^\circ(x) = g(x)$.
- For all $d : s \to \coprod_{i \in I} e_i \in D$ and $t = (t_i)_{i \in I} \in \bigtimes_{i \in I} \in \overline{T_\Sigma}(V)_{e_i}$,

$$g^\circ(d(t)) = [g^\circ(t_i)]_{i \in I} \circ d^{\mathcal{A}}.$$

- For all $e : \prod_{i\in I}\coprod_{j\in J} e_{ij} \in \mathcal{T}_{po}(S)$, $i \in I$ and $t = (t_j)_{j\in J} \in \times_{j\in J}\overline{T_\Sigma}(V)_{e_{ij}}$,

$$g^\circ(i(t)) = [g^\circ(t_j)]_{j\in J} \circ \pi_i.$$

The flowchart valuation $\eta_V : V \to A_V$ is defined as follows:

For all $x : s \in V$ and $a \in A_s$, $\eta_V(x)(a) = (x, a)$.

Given $e \in \mathcal{T}_{po}(S)$ and $t, t' \in \overline{T_\Sigma}(V)_e$, $\mathcal{A}$ **satisfies** the (**flowchart**) **equation** $t = t'$ in Σ-algebra $\mathcal{A}$ with carrier A, written as $\mathcal{A} \models t = t'$, if $\eta_V^\circ(t) = \eta_V^\circ(t')$.

Lemma 9.19 (Substitutionslemma)

Let $\Sigma, V, O, \mathcal{A}, A$ be as above. For all flowchart substitutions $g : V \to \overline{T_\Sigma}(O)$ and flowchart valuations $h : O \to A_O$,

$$(h^\circ \circ g)^\circ = h^\circ \circ g^*.$$

Proof. Let $t \in \overline{T_\Sigma}(V)$. We show

$$(h^\circ \circ g)^\circ(t) = h^\circ(g^*(t)) \qquad (1)$$

by induction on t.

Case 1. $t \in V$. Then $(h^\circ \circ g)^\circ(t) = (h^\circ \circ g)(t) = h^\circ(g(t)) = h^\circ(g^*(t))$.

Case 2. $t = d(u)$ for some $d : s \to e \in D$ and $u \in \overline{T_\Sigma}(V)$. Then

$$(h^\circ \circ g)^\circ(t) = (h^\circ \circ g)^\circ(u) \circ d^{\mathcal{A}} \stackrel{ind.\ hyp.}{=} h^\circ(g^*(u)) \circ d^{\mathcal{A}} = h^\circ(d(g^*(u)))$$
$$= h^\circ(g^*(d(u))) = h^\circ(g^*(t)).$$

Case 3. $t = i(u)$ for some $i \in I$, $I \in \mathcal{I}$, and $u = (t_j)_{j\in J} \in \bigtimes_{j\in J}\overline{T_\Sigma}(V)_{e_{ij}}$. Then

$$(h^\circ \circ g)^\circ(t) = [(h^\circ \circ g)^\circ(t_j)]_{j\in J} \circ \pi_i \stackrel{ind.\ hyp.}{=} [h^\circ(g^*(t_j))]_{j\in J} \circ \pi_i$$
$$= h^\circ(i((g^*(t_j))_{j\in J})) = h^\circ(i(g^*(u))) = h^\circ(g^*(i(u))) = h^\circ(g^*(t)).$$

❏

9.20 From flowcharts to terms

Destructive polynomial signatures $\Sigma = (S, F)$ generalize signatures for *comodels* and *effectful programming* (see, e.g., [145, 140, 27]). They have constructive counterparts that admit interpretations as state transitions.

To be more precise, let

$$\begin{aligned}
\overline{S} &= S \cup \{\overline{I} \mid I \subseteq \mathcal{I}\} \cup \{\overline{e} \mid e = \prod_{i\in I}\coprod_{j\in J} e_{ij} \in \mathcal{T}_{po}(S)\},\\
\overline{F} &= \{\overline{d} : \prod_{i\in I} e_i \to s \mid d : s \to \coprod_{i\in I} e_i \in D\} \cup \\
&\quad \{\overline{i} : \prod_{j\in J} e_{ij} \to \overline{e} \mid e = \prod_{i\in I}\coprod_{j\in J} e_{ij} \in \mathcal{T}_{po}(S),\ i \in I\}.
\end{aligned}$$

Hence $\overline{\Sigma} = (S, \overline{F})$ is constructive. A Σ-algebra $\mathcal{A}$ induces the $\overline{\Sigma}$-algebra $\mathcal{A}_V$ that is defined as follows:

Let A be the carrier of $\mathcal{A}$ and A_V be the $\mathcal{T}_{po}(S)$-sorted set defined in section 9.19.

- For all $s \in S$, $\mathcal{A}_V(s) = A_{V,s}$.
- For all $s \in \overline{S} \setminus S$, $\mathcal{A}_V(\overline{s}) = A_{V,s}$.
- For all $d : s \to \coprod_{i\in I} e_i \in D$ and $f = (f_i)_{i\in I} \in \times_{i\in I} A_{V,e_i}$,

$$\overline{d}^{\mathcal{A}_V}(f) = [f_i]_{i\in I} \circ d^{\mathcal{A}} : A_{V,s}.$$

- For all $e = \prod_{i\in I}\coprod_{j\in J} e_{ij} \in \mathcal{T}_{po}(S)$, $i \in I$ and $f = (f_j)_{j\in J} \in \times_{j\in J} A_{V,e_{ij}}$,

$$\overline{i}^{\mathcal{A}_V}(f) = [f_j]_{j\in J} \circ \pi_i : A_{V,e}.$$

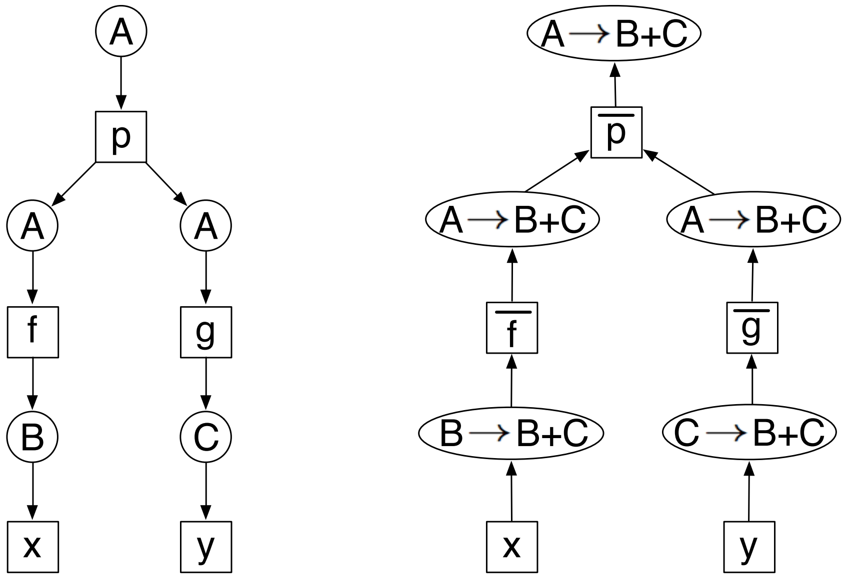
Hence $\overline{d}^{\mathcal{A}_V} = \mathit{lift}_{V\times A}(d^{\mathcal{A}})$ and $\overline{i}^{\mathcal{A}_V} = \mathit{lift}_{V\times A}(\pi_i)$ (see section 2.4).

Given a Σ-flowchart t, the $\overline{\Sigma}$-term $\overline{t}$ is obtained from t by replacing every node label $d \in D$ with $\overline{d}$.

If $t : s \in \overline{T_\Sigma}(V)$ for some $s \in S$, then for all subflowcharts $u : e$ of t, $e \in S$.

Example

Here are a Σ-flowchart and its corresponding $\overline{\Sigma}$-term, extended (in ovals) by the source and target types of the involved destructors and constructors, respectively.



Let $\varphi : A \to 2$ and $p : A \to A + A$ be defined as follows: $p(a) = \iota_1(a) \Leftrightarrow \varphi(a) = 1$. Moreover, let $f : A \to B$, $g : A \to C$ and $V = \{x, y\}$.

Then for all $h_1, h_2 : A \to B + C$, $h_3 : B \to B + C$, $h_4 : C \to B + C$ and $a \in A$,

$$\overline{p}(h_1, h_2)(a) = ([h_1, h_2] \circ p)(a) = [h_1, h_2](p(a)) = \left\{ \begin{array}{ll} h_1(a) & \text{if } p(a) = \iota_1(a) \\ h_1(a) & \text{if } p(a) = \iota_2(a) \end{array} \right\}$$

$$= \left\{ \begin{array}{ll} h_1(a) & \text{if } \varphi(a) = 1 \\ h_1(a) & \text{if } \varphi(a) = 0 \end{array} \right\},$$

$$\overline{f}(h_3)(a) = (h_3 \circ f)(a) = h_3(f(a)), \ \overline{g}(h_4)(a) = (h_4 \circ g)(a) = h_4(g(a)). \quad ❏$$

More interesting examples involve cycles, expressed in terms of flowchart equations (see section 17.5).

A_V yields a monad

$\eta_V : V \to A_V$ is an instance of the unit $\eta : Id_{Set^S} \to T$ of the Set^S-sorted version

$$M = (T : Set^S \to Set^S, \eta, \mu)$$

of the state monad defined in section 24.1:

Given an S-sorted set A, T maps a S-sorted set V to the S-sorted set A_V (see above) and an S-sorted function $f : V \to V'$ to

$$T(f) = \lambda g.\lambda(x, a).(f(x), a) \circ g : T(V) \to T(V')$$

(see section 5.1).

For obtaining **Lemma 9.20** below, we make use of the extension operator $\_^*$ of the Kleisli triple induced by M (see chapter 24).

More precisely, we need the instance $\_^+ : (V \to TV) \to (TV \to TV)$ that satisfies the equation

$$f^+ \circ \eta_V = f \tag{1}$$

for all $f : V \to TV$ and is defined as follows (analogously to the state functor; see section 24.1): For all $e \in \mathcal{T}_{po}(S)$,

$$\begin{array}{rcl} f_e^+ : A_{V,e} & \to & A_{V,e} \\ g & \mapsto & \lambda a.f(\pi_1(ga))(\pi_2(ga)). \end{array}$$

Indeed, (1) holds true:

For all $x : s \in V$ and $a \in A_s$,

$$(f^+ \circ \eta_V)(x)(a) = f^+(\eta_V(x))(a) = f(\pi_1(\eta_V(x)(a)))(\pi_2(\eta_V(x)(a))) = f(x)(a).$$

In section 9.19, S-sorted functions $f : V \to TV$ were called flowchart valuations.

By (1), their term extensions $f^* : \overline{T_\Sigma}(V) \to A_V$ can be reduced to η_V^*—provided that $f^+ : A_V \to A_V$ is $\overline{\Sigma}$-homomorphic:

$$f^* \stackrel{(1)}{=} (f^+ \circ \eta_V)^* \stackrel{Lemma\ 9.9}{=} f^+ \circ \eta_V^*. \tag{2}$$

Lemma 9.21

For all flowchart valuations $f : V \to A_V$, $f^+ : A_V \to A_V$ is $\overline{\Sigma}$-homomorphic, i.e., for all $d : s \to \coprod_{i\in I} e_i \in D$ and $g = (g_i)_{i\in I} \in \bigtimes_{i\in I} A_{V,e_i}$,

$$\overline{d}^{\mathcal{A}_V}(f^+_{e_i}(g_i))_{i\in I} = f^+_s(\overline{d}^{\mathcal{A}_V}(g)), \tag{3}$$

and for all $e = \prod_{i\in I} \coprod_{j\in J} e_{ij} \in \mathcal{T}_{po}(S)$, $i \in I$ and $g = (g_j)_{j\in J} \in \bigtimes_{j\in J} A_{V,e_{ij}}$,

$$\overline{i}^{\mathcal{A}_V}(f^+_{e_{ij}}(g_j))_{j\in J} = f^+_e(\overline{i}^{\mathcal{A}_V}(g)). \tag{4}$$

Proof. (3): Let $a \in A_s$, $k \in I$, $b \in A_{e_k}$ and $x \in V$ such that $d^{\mathcal{A}}(a) = \iota_k(b)$. Then

$$\overline{d}^{\mathcal{A}_V}(f^+_{e_i}(g_i))_{i\in I}(a) = [f^+_{e_i}(g_i)]_{i\in I}(d^{\mathcal{A}}(a)) = [f^+_{e_i}(g_i)]_{i\in I}(\iota_k(b)) = f^+_{e_k}(g_k)(b)$$
$$= f(\pi_1(g_k(b)))(\pi_2(g_k(b))) = f(\pi_1([g_i]_{i\in I}(\iota_k(b))))(\pi_2([g_i]_{i\in I}(\iota_k(b))))$$
$$= f(\pi_1([g_i]_{i\in I}(d^{\mathcal{A}}(a))))(\pi_2([g_i]_{i\in I}(d^{\mathcal{A}}(a)))) = f(\pi_1(([g_i]_{i\in I} \circ d^{\mathcal{A}})(a)))(\pi_2(([g_i]_{i\in I} \circ d^{\mathcal{A}})(a))) =$$
$$= f^+_s(\overline{d}^{\mathcal{A}_V}(g))(a).$$

(4): Let $a \in A_e$, $k \in J$, $b \in A_{e_{ik}}$ and $x \in V$ such that $\pi_i^{\mathcal{A}}(a) = \iota_k(b)$. Then

$$\overline{i}^{\mathcal{A}_V}(f^+_{e_{ij}}(g_j))_{j\in J}(a) = [f^+_{e_{ij}}(g_j)]_{j\in J}(\pi_i(a)) = [f^+_{e_{ij}}(g_{ij})]_{j\in J}(\iota_k(b)) = f^+_{e_{ik}}(g_k)(b)$$
$$= f(\pi_1(g_k(b)))(\pi_2(g_k(b))) = f(\pi_1([g_j]_{j\in J}(\iota_k(b))))(\pi_2([g_j]_{j\in J}(\iota_k(b))))$$
$$= f(\pi_1([g_j]_{j\in J}(\pi_i(a))))(\pi_2([g_j]_{j\in J}(\pi_i(a))))$$
$$= f(\pi_1(([g_j]_{j\in J} \circ \pi_i)(a)))(\pi_2(([g_j]_{j\in J} \circ \pi_i)(a))) = f^+_e([g_j]_{j\in J} \circ \pi_i)(a)$$
$$= f^+_e(\pi_i^{\mathcal{A}_V}(g))(a).$$

❏

Theorem 9.22 (Destructive signatures and their constructive counterparts satisfy the same equations)

Let $\Sigma = (S, F)$ be a destructive signature and $\overline{\Sigma}$, $\mathcal{A}$ and $\mathcal{A}_V$ be as above.

For all $t \in \overline{T_\Sigma}(V)$, $\eta_V^*(\overline{t}) = \eta_V^\circ(t)$. (5)

For all $s \in S$ and $t, t' \in \overline{T_\Sigma}(V)_s$, $\mathcal{A} \models t = t'$ iff $\mathcal{A}_V \models \overline{t} = \overline{t'}$. (6)

Proof of (5) by induction on t.

For all $x : s \in V$, $\eta_V^*(\overline{x}) = \eta_V(x) = \eta_V^\circ(x)$.

For all $d : s \to \coprod_{i \in I} e_i \in D$ and $t = (t_i)_{i \in I} \in \bigtimes_{i \in I} \overline{T_\Sigma}(V)_{e_i}$,

$$\eta_V^*(\overline{d(t)}) = \eta_V^*(\overline{d}(\overline{t})) = \overline{d}^{\mathcal{A}_V}(\eta_V^*(\overline{t})) = \overline{d}^{\mathcal{A}_V}(\eta_V^*(\overline{t_i}))_{i \in I} = [\eta_V^*(\overline{t_i})]_{i \in I} \circ d^{\mathcal{A}}$$
$$\stackrel{ind.\ hyp.}{=} [\eta_V^\circ(t_i)]_{i \in I} \circ d^{\mathcal{A}} = \eta_V^\circ(d(t)).$$

For all $e : \prod_{i \in I} \coprod_{j \in J} e_{ij} \in \mathcal{T}_{po}(S)$, $i \in I$ and $t = (t_j)_{j \in J} \in \bigtimes_{j \in J} \overline{T_\Sigma}(V)_{e_{ij}}$,

$$\eta_V^*(\overline{i(t)}) = \eta_V^*(\overline{i}(\overline{t})) = \overline{i}^{\mathcal{A}_V}(\eta_V^*(\overline{t})) = \overline{i}^{\mathcal{A}_V}(\eta_V^*(\overline{t_j}))_{j \in J} = [\eta_V^*(\overline{t_j})]_{j \in J} \circ \pi_i$$
$$\stackrel{ind.\ hyp.}{=} [\eta_V^\circ(t_j)]_{j \in J} \circ \pi_i = \eta_V^\circ(i(t)).$$

Proof of (6).

Let $s \in S$, $t, t' \in \overline{T_\Sigma}(V)_s$ and $f : V \to A_V$ be an S-sorted function such that $\mathcal{A}$ satisfies $t = t'$, i.e., $\eta_V^\circ(t) = \eta_V^\circ(t')$. Then

$$f^*(\overline{t}) \overset{(2)}{=} f^+(\eta_V^*(\overline{t})) \overset{(5)}{=} f^+(\eta_V^\circ(t)) = f^+(\eta_V^\circ(t')) \overset{(5)}{=} f^+(\eta_V^*(\overline{t'})) \overset{(2)}{=} f^*(\overline{t'}).$$

Conversely, suppose that $\mathcal{A}_V$ satisfies $\overline{t} = \overline{t'}$. Then for all S-sorted functions $f : V \to A_V$, $f^*(\overline{t}) = f^*(\overline{t'})$, in particular, $\eta_V^\circ(t) = \eta_V^*(\overline{t}) = \eta_V^*(\overline{t'}) = \eta_V^\circ(t')$. ❏

Let $\Sigma = (S, F)$ be a signature that includes *KripkeSig* (see section 8.3).

This condition allows us to specify almost any kind of Kripke-like structure and express as Σ-formulas not only higher-order functions, λ-expressions, etc., but also logic operators known from λProlog, CTL (computation tree logic), the modal μ-calculus, query languages like SQL, relational algebra, description logic or XPath (see, e.g., [110, 115, 58, 164, 105]) and dynamic logic, which admits the verification of imperative programs with iteration and has been reformulated coalgebraically in [114], 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 quantifiers used in description logic, while the "advanced" CTL operators can be reduced to fixpoint operators known from the modal μ-calculus.

While every formula of CTL or the modal μ-calculus denotes 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.

Σ-formulas are built upon $\lambda\Sigma$-terms (see below), which admit the highly structured specification of, e.g., states, labels, atoms as well as transition and valuation functions (see above) by subtle case distinctions based on pattern matching. Morever, 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 Σ with suitable (polynomial or weighted) destructors.

In contrast to Σ-arrows (see chapter 8), Σ-formulas may contain variables of any type over S. Whereas fixpoint operators occurring in Σ-formulas always bind a single variable of a powerset type, λ-abstraction occurring in Σ-formulas may bind several variables of any type of $\mathcal{T}(S)$. Of course, classical (many-sorted) first-order logic with single-variable quantification is also included.

Σ-terms and Σ-flowcharts also contain variables. Σ-terms have *polynomial* types and are composed of constructors and thus always denote (parametrized) *objects* of initial algebras (see chapter 9).

Hence a Σ-term is *both* a particular Σ-formula φ *and* the result of interpreting φ in an initial term algebra. Σ-flowcharts represent *functions* into sum types, which may also be specified as Σ-arrows or (variable-free) Σ-formulas. Variables of flowcharts denote exits for output, not place-holders for input as Σ-terms and Σ-formulas do.

Finally, Σ-*co*terms denote *behaviours*, which are—like Σ-terms—particular *object* representations. But—in contrast to Σ-terms—they have no *functional* interpretation.

10.1 Syntax

Let V be a $\mathcal{T}(S)$-sorted set of variables,

$$trace = (state + label \times state)^*, \quad row = (state + 1)^{label}, \; table = \mathcal{P}(row).$$

The $\mathcal{T}(S)$-sorted set $\Lambda_\Sigma(V)$ of **λΣ-terms over** V is inductively defined as follows:

Let $e, e' \in \mathcal{T}(S)$.

- $Arr_\Sigma \cup T_\Sigma(V) \subseteq \Lambda_\Sigma(V)$. (Σ-arrows and well-founded Σ-terms)

- For all $I \subseteq \mathcal{I}$ and $(f_i : e_i \to e)_{i \in I} \in \Lambda_\Sigma(V)^I$, (sum extension)

$$[f_i]_{i \in I} : \coprod_{i \in I} e_i \to e \in \Lambda_\Sigma(V).$$

- For all $I \subseteq \mathcal{I}$ and $i \in I$, $\iota_i : e_i \to \coprod_{i \in I} \in \Lambda_\Sigma(V)$. (injection)
- $true : \mathcal{P}(e) \in \Lambda_\Sigma(V)$.
- $\neg : \mathcal{P}(e) \to \mathcal{P}(e) \in \Lambda_\Sigma(V)$. (complement)
- $\wedge : \mathcal{P}(e) \times \mathcal{P}(e) \to \mathcal{P}(e) \in \Lambda_\Sigma(V)$. (intersection)
- $\neg : 2 \to 2 \in \Lambda_\Sigma(V)$. (negation)
- $\wedge : 2 \times 2 \to 2 \in \Lambda_\Sigma(V)$. (conjunction)
- $ite : 2 \times e \times e \to e \in \Lambda_\Sigma(V)$. (conditional)
- $(=) : \mathcal{P}(e \times e) \in \Lambda_\Sigma(V)$. (equality)
- For all $I \subseteq \mathcal{I}$ and $f : \prod_{i \in I} e_i \to e,\ t = (t_i : e_i)_{i \in I} \in \Lambda_\Sigma(V)^I$, $f(t) : e \in \Lambda_\Sigma(V)$. (application)
- For all $x : e \in V$ and $t : e' \in \Lambda_\Sigma(V)$, ($\lambda$-abstraction)

$$\lambda x.t : e \to e' \in \Lambda_\Sigma(V).$$

- $single : e \to \mathcal{P}(e) \in \Lambda_\Sigma(V)$. (singleton)

- $card : \mathcal{P}(e) \to \mathbb{N} \cup \{\omega\} \in \Lambda_\Sigma(V)$ (cardinality)
- $map : (e \to e') \to \mathcal{P}(e) \to \mathcal{P}(e') \in \Lambda_\Sigma(V)$.
- $filter : (e \to 2) \to \mathcal{P}(e) \to \mathcal{P}(e) \in \Lambda_\Sigma(V)$. (selection)
- $join, meet : \mathcal{P}(\mathcal{P}(e)) \to \mathcal{P}(e) \in \Lambda_\Sigma(V)$.
- $foldl : (e \to e' \to e) \to e \to e'^* \to e \in \Lambda_\Sigma(V)$. (list folding)
- $foldNDS : (e \to \mathcal{P}(e)) \to (e \to e' \to \mathcal{P}(e)) \to e \to e'^* \to \mathcal{P}(e) \in \Lambda_\Sigma(V)$.
  (nondeterministic list folding including silent transitions)
- $\chi : \mathcal{P}(e) \to e \to 2 \in \Lambda_\Sigma(V)$. (characteristic function)
- $rel2fun : \mathcal{P}(e \times e') \to e \to \mathcal{P}(e') \in \Lambda_\Sigma(V)$. (relation-to-function transformer)
- $inv : \mathcal{P}(e \times e') \to \mathcal{P}(e' \times e) \in \Lambda_\Sigma(V)$. (inverse relation)
- $\pi_1 : \mathcal{P}(e \times e') \to \mathcal{P}(e),\ \pi_2 : \mathcal{P}(e \times e') \to \mathcal{P}(e') \in \Lambda_\Sigma(V)$. (set projection)
- $(*) : \mathcal{P}(e) \times \mathcal{P}(e') \to \mathcal{P}(e \times e') \in \Lambda_\Sigma(V)$. (Cartesian product)
- $(/),\ \overline{\forall} : \mathcal{P}(e \times e') \to \mathcal{P}(e') \to \mathcal{P}(e) \in \Lambda_\Sigma(V)$.
  (relational division, universal projection)
- $(;) : \mathcal{P}(e \times e') \times \mathcal{P}(e' \times e'') \to \mathcal{P}(e \times e'') \in \Lambda_\Sigma(V)$. (relational composition)
- For all $x : \mathcal{P}(e) \in V$ and $\varphi : \mathcal{P}(e) \in \Lambda_\Sigma(V)$, $\mu x.\varphi : \mathcal{P}(e) \in \Lambda_\Sigma(V)$. ($\mu$-abstraction)
- For all $x : e \in V$ and $\varphi : e,\ \psi : e' \in \Lambda_\Sigma(V)$, $\psi[\varphi/x] \in \Lambda_\Sigma(V)$. (substitution)

- $\sim : \mathcal{P}(state^2) \in \Lambda_\Sigma(V)$. (behavioral equivalence)
- $labels : \mathcal{P}(label) \in \Lambda_\Sigma(V)$.
- $preds : state \to \mathcal{P}(state) \in \Lambda_\Sigma(V)$,
  $predsL : state \to label \to \mathcal{P}(state) \in \Lambda_\Sigma(V)$. (predecessors)
- $out : state \to \mathcal{P}(atom) \in \Lambda_\Sigma(V)$,
  $outL : state \times label \to \mathcal{P}(atom) \in \Lambda_\Sigma(V)$, (output)
- $child : \mathcal{P}(label) \to \mathcal{P}(state^2) \in \Lambda_\Sigma(V)$.
- $traces : state \to state \to \mathcal{P}(trace) \in \Lambda_\Sigma(V)$,
- $transL2row : state \to row \in \Lambda_\Sigma(V)$.
- $+ : row \times row \to row \in \Lambda_\Sigma(V)$.
- $projectrow : \mathcal{P}(label) \to row \to row \in \Lambda_\Sigma(V)$.
- $selectrow : (label \to state) \to row \to 2 \in \Lambda_\Sigma(V)$.
- $labs : table \to \mathcal{P}(label) \in \Lambda_\Sigma(V)$.
- $\wedge,\ *,\ / : table \times table \to table \in \Lambda_\Sigma(V)$.
- $njoin : table \to table \to table \in \Lambda_\Sigma(V)$.
- $fundeps : table \to \mathcal{P}(label \times \mathcal{P}(label)) \in \Lambda_\Sigma(V)$. (functional dependencies)

A $\lambda\Sigma$-term t over V is **flat** if there are $f : \prod_{i\in I} e_i \to \prod_{j\in J} e'_j \in F$ and $x = (x_i)_{i\in I} \in V^I$ such that $t = f(x)$ or $t = \pi_j(f(x))$ for some $j \in J$.

The set $Fo_\Sigma(V)$ of (first-order) **Σ-formulas over** V is inductively defined as follows:

- $True \in Fo_\Sigma(V)$.
- For all $p : \mathcal{P}(e),\ t : e \in \Lambda_\Sigma(V)$, $p(t) \in Fo_\Sigma(V)$. (Σ-atoms)
- In particular, for all $t, t' : e \in \Lambda_\Sigma(V)$, $t = t' \in Fo_\Sigma(V)$. (Σ-equations)
- For all $\varphi \in Fo_\Sigma(V)$, $\neg\varphi \in Fo_\Sigma(V)$. (negation)
- For all $\varphi, \psi \in Fo_\Sigma(V)$, $\varphi \wedge \psi \in Fo_\Sigma(V)$. (conjunction)
- For all $x \in V$ and $\varphi \in Fo_\Sigma(V)$, $\forall x\varphi \in Fo_\Sigma(V)$. (universal quantification)

10.2 Derived terms and formulas

Type instances

For all $\mathcal{F} = \{f_s : e_s \to e'_s \mid s \in S\} \subseteq \Lambda_\Sigma(V)$ and $e \in \mathcal{T}_{po}(S)$,

$$\mathcal{F}_e : e[e_s/s \mid s \in S] \to e[e'_s/s \mid s \in S]$$

is inductively defined as follows:

For all $s \in S$, $I \subseteq \mathcal{I}$ and $(e_i)_{i\in I} \in \mathcal{T}_{po}(S)^I$, $e \in \mathcal{T}_{po}(S)$, commutative monoids $(M, +, 0)$ and $C \subseteq M$,

$$\mathcal{F}_s = f_s, \quad \mathcal{F}_1 = id_1, \quad \mathcal{F}_{\prod_{i\in I} e_i} = \prod_{i\in I} \mathcal{F}_{e_i}, \quad \mathcal{F}_{\coprod_{i\in I} e_i} = \coprod_{i\in I} \mathcal{F}_{e_i},$$
$$\mathcal{F}_{(M\times e)_C} = (id_M \times \mathcal{F}_e)_C.$$

Further derived $\lambda\Sigma$-terms

- For all $e \in \mathcal{T}(S)$, $id_e = \lambda x.x : e \to e$. (identities)
- For all $n > 1$, $x = (x_1 : e_1, \dots, x_n : e_n) \in V^n$ and $t : e \in \Lambda_\Sigma(V)$, (uncurrying)

$$\lambda(x_1, \dots, x_n).t = \lambda x_1 \dots \lambda x_n.t : e_1 \times \dots \times e_n \to e.$$

- $\$ = \lambda(f, x).\lambda x.f(x) : (e \to e') \times e \to e'$. (application)
- $false = \neg true : \mathcal{P}(e)$.
- $(\neq) = \neg \circ (=) : e \times e \to 2$. (inequality)
- $sat = \chi^{-1} : (e \to 2) \to \mathcal{P}(e)$. (satisfying elements)

- For all $e' \in \{2, \mathcal{P}(e)\}$,

$$\begin{aligned}
\vee &= \lambda(x,y).\neg(\neg x \wedge \neg y) : e' \times e' \to e', \\
\Rightarrow &= \lambda(x,y).\neg x \vee y : e' \times e' \to e', \\
\ominus &= \lambda(x,y).x \wedge \neg y : e' \times e' \to e', \\
\Leftarrow &= \lambda(x,y).y \Rightarrow x : e' \times e' \to e', \\
\Leftrightarrow &= \lambda(x,y).(x \Rightarrow y) \wedge (x \Leftarrow y) : e' \times e' \to e', \\
\oplus &= \lambda(x,y).(x \ominus y) \vee (y \ominus x) : e' \times e' \to e'.
\end{aligned}$$

- $\circ = \lambda(f,g).\lambda x.g(f(x)) : (e \to e') \times (e' \to e'') \to e \to e''$. (composition)
- For all $I \subseteq \mathcal{I}$, $i \in I$ and $(f_i : e \to e_i)_{i \in I} \in \Lambda_\Sigma(V)^I$, (product extension)

$$\langle f_i \rangle_{i \in I} = \lambda x.(t_i(x))_{i \in I} : e \to \prod_{i \in I} e_i.$$

- For all $I \subseteq \mathcal{I}$ and $i \in I$, $\pi_i = \lambda(x_i)_{i \in I}.x_i : \prod_{i \in I} \to e_i$. (functional projection)
- For all $I \subseteq \mathcal{I}$, $i \in I$ and $(f_i : e_i \to e'_i)_{i \in I} \in \Lambda_\Sigma(V)^I$, (product, sum)

$$\begin{aligned}
\textstyle\prod_{i \in I} f_i &= \langle f_i \circ \pi_i \rangle_{i \in I} : \textstyle\prod_{i \in I} e_i \to \prod_{i \in I} e'_i, \\
\textstyle\coprod_{i \in I} f_i &= [\iota_i \circ f_i]_{i \in I} : \textstyle\coprod_{i \in I} e_i \to \coprod_{i \in I} e'_i.
\end{aligned}$$

- For all $f : e \to \mathcal{P}(e'),\ g : e' \to \mathcal{P}(e''),\ \varphi : \mathcal{P}(e) \in \Lambda_\Sigma(V)$, (monadic operators)

$$\begin{aligned} \mathit{joinMap}(f) &= \mathit{join} \circ \mathit{map}(f) : \mathcal{P}(e) \to \mathcal{P}(e'), \\ g \mathrel{<\!\!=\!\!<} f &= \mathit{joinMap}(g) \circ f : e \to \mathcal{P}(e''), \\ \varphi \mathrel{>\!\!>\!\!=} f &= \mathit{joinMap}(f)(\varphi) : \mathcal{P}(e'). \end{aligned}$$

- For all $f : e \to e' \to e'' \in \Lambda_\Sigma(V)$, (flip)

$$\mathit{flip}(f) = \lambda y.\lambda x.f(x)(y) : e' \to e \to e''.$$

- For all $n > 1$ and $f : \prod_{i=1}^{n} e_i \to e \in \Lambda_\Sigma(V)$, (currying)

$$\mathit{curry}(f) = \lambda x_1 \ldots \lambda x_n.f(x_1, \ldots, x_n) : e_1 \to \cdots \to e_n \to e.$$

- For all $n \in \mathbb{N}$, $t : e \in \Lambda_\Sigma(V)$ and $(t_i : e_i)_{i=1}^{n} \in \Lambda_\Sigma(V)^n$,

$$(t \ \mathit{where} \ x_1 = t_1; \ldots; x_n = t_n) = (\lambda(x_1, \ldots, x_n).t)(t_1, \ldots, t_n) : e.$$

- For all $r : \mathcal{P}(e \times e'),\ \psi : \mathcal{P}(e') \in \Lambda_\Sigma(V)$, $\overline{\exists}(r)(\psi) = \neg\overline{\forall}(r)(\neg\psi) : \mathcal{P}(e)$. (existential projection)

- For all $\varphi : \mathcal{P}(state)$, $st : state \in \Lambda_\Sigma(V)$ and $x : \mathcal{P}(state)$, $z : label \in V$,

$$\begin{aligned}
trans^*(\varphi) &= \mu x.(\varphi \vee joinMap(trans)(x)) : \mathcal{P}(state),\\
transL^*(\varphi) &= \mu x.(\varphi \vee joinMap(\lambda z.joinMap(transL)(x))(labels))\\
&\qquad : \mathcal{P}(state),\\
transAll(\varphi) &= trans^*(\varphi) \vee transL^*(\varphi) \qquad : \mathcal{P}(state),\\
unfoldNDS(st) &= out <\!\!=\!\!< foldNDS(trans^*)(transL)(st)\\
&\qquad : label^* \rightarrow \mathcal{P}(atom),\\
unfoldND(st) &= out <\!\!=\!\!< foldNDS(single)(transL)(st)\\
&\qquad : label^* \rightarrow \mathcal{P}(atom),\\
unfoldD(st) &= out \circ foldl(transL)(st) : label^* \rightarrow \mathcal{P}(atom)\\
&\qquad : label^* \rightarrow \mathcal{P}(atom).
\end{aligned}$$

  $unfoldD(st)$ can be interpreted $\mathcal{A} \in Alg_\Sigma$ only if for all $a \in A_{state}$ and $lab \in A_{label}$, $|transL^{\mathcal{A}}(a)(lab)| = 1$.

- For all $ts : \mathcal{P}(label) \in \Lambda_\Sigma(V)$, $sibling(ts) = child(ts); parent(ts) : \mathcal{P}(state^2)$.
- For all $\varphi : \mathcal{P}(e)$, $\psi : \mathcal{P}(e')$, $r : \mathcal{P}(e \times e') \in \Lambda_\Sigma(V)$,

$$\varphi \ll r = \varphi \wedge \pi_1 r : \mathcal{P}(e) \quad \text{and} \quad r \gg \psi = \pi_2 r \wedge \psi : \mathcal{P}(e').$$

- $parent = inv \circ child : \mathcal{P}(label) \to \mathcal{P}(state^2)$.
- For all $ts : \mathcal{P}(label),\ t : label \in \Lambda_\Sigma(V)$,

$$
\begin{aligned}
[ts] &= \overline{\forall}\ child(ts) : \mathcal{P}(state) \to \mathcal{P}(state),\\
\langle ts\rangle &= \overline{\exists}\ child(ts) : \mathcal{P}(state) \to \mathcal{P}(state),\\
[t] &= [single(t)] : \mathcal{P}(state) \to \mathcal{P}(state),\\
\langle t\rangle &= \langle single(t)\rangle : \mathcal{P}(state) \to \mathcal{P}(state),\\
\Box &= [\emptyset] : \mathcal{P}(state) \to \mathcal{P}(state),\\
\Diamond &= \langle\emptyset\rangle : \mathcal{P}(state) \to \mathcal{P}(state).
\end{aligned}
$$

- For all $r : \mathcal{P}(e^2) \in \Lambda_\Sigma(V)$ and $x : \mathcal{P}(e^2) \in V$, (transitive closure)

$$tcl(r) = \mu x.(r \vee (r; x)) : \mathcal{P}(e^2).$$

- For all $ts : \mathcal{P}(label) \in \Lambda_\Sigma(V)$,

$$
\begin{aligned}
descendant(ts) &= tcl(child(ts)) : \mathcal{P}(state^2),\\
ancestor(ts) &= tcl(parent(ts)) : \mathcal{P}(state^2).
\end{aligned}
$$

- For all $ts : \mathcal{P}(label) \in \Lambda_\Sigma(V)$,

$$related(ts) = ancestor(ts); sibling(ts); descendant(ts) : \mathcal{P}(state^2).$$

- For all $x : \mathcal{P}(e) \in V$ and negation-free $\varphi : \mathcal{P}(e) \in \Lambda_\Sigma(V)$, ($\nu$-abstraction)

$$\nu x.\varphi = \neg\mu x.\neg\varphi[\neg x/x] : \mathcal{P}(e).$$

- For all $\varphi, \psi : \mathcal{P}(state) \in \Lambda_\Sigma(V)$ and some $x : \mathcal{P}(state) \in V$,

$$\begin{aligned}
EF(\varphi) &= \mu x.(\varphi \vee \Diamond x) \\
&= \exists(descendant(\emptyset))(\varphi) : \mathcal{P}(state), && (\varphi \text{ finally on some } child\text{-path}) \\
AG(\varphi) &= \nu x.(\varphi \wedge \Box x) \\
&= \forall(descendant(\emptyset))(\varphi) : \mathcal{P}(state), && (\varphi \text{ generally on all } child\text{-paths}) \\
EG(\varphi) &= \nu x.(\varphi \wedge (\Diamond x \vee \Box false)) : \mathcal{P}(state), && (\varphi \text{ generally on some } child\text{-path}) \\
AF(\varphi) &= \mu x.(\varphi \vee (\Box x \wedge \Diamond True)) : \mathcal{P}(state), && (\varphi \text{ finally on all } child\text{-paths}) \\
\varphi EU \psi &= \mu x.(\psi \vee (\varphi \wedge \Diamond x)) : \mathcal{P}(state), && (\varphi \text{ until } \psi \text{ on some } child\text{-path}) \\
\varphi AU \psi &= \mu x.(\psi \vee (\varphi \wedge \Box x)) : \mathcal{P}(state). && (\varphi \text{ until } \psi \text{ on all } child\text{-paths})
\end{aligned}$$

- $$\begin{aligned}
trans2tab &= map(transL2row) \circ trans : state \rightarrow table, \\
project &= map \circ projectrow : table \rightarrow table, \\
select &= filter \circ selectrow : table \rightarrow table.
\end{aligned}$$

- For all $f : label \rightarrow state \in \Lambda_\Sigma(V)$ and $\vartheta, \vartheta' : table \in \Lambda_\Sigma(V)$,

$$\begin{aligned} \vartheta \vee \vartheta' &= \neg(\neg\vartheta \wedge \neg\vartheta'), \\ \vartheta - \vartheta' &= \vartheta \wedge \neg\vartheta', \\ tjoin(f)(\vartheta) &= select(f) \circ njoin(\vartheta) : table \rightarrow table. \end{aligned}$$

Derived Σ-formulas

- $False = \neg\, True$.
- For all $\varphi, \psi \in Fo_\Sigma(V)$,

$$\begin{aligned} \varphi \vee \psi &= \neg(\neg\varphi \wedge \neg\psi), \\ \varphi \Rightarrow \psi &= \neg\varphi \vee \psi, \\ \varphi \ominus \psi &= \varphi \wedge \neg\psi, \\ \varphi \Leftrightarrow \psi &= (\varphi \Rightarrow \psi) \wedge (\varphi \Rightarrow \psi), \\ \varphi \oplus \psi &= (\varphi \ominus \psi) \vee (\psi \ominus \varphi). \end{aligned}$$

- For all $I \in \mathcal{I}$, $x = (x_i : e : i)_{i \in I} \in V^I$ and $\varphi \in Fo_\Sigma(V)$, $\exists x\varphi = \neg\forall x\neg\varphi$.

(existential quantification)

Every Σ-formula or $\lambda\Sigma$-term t yields a labelled tree over $(\mathbb{N}, Op)$ (see section 2.9) where Op is the set of the constants, function symbols and variables t is composed of. Higher-order terms $t(t_1)\dots(t_n)$ are turned into their first-order counterparts $(\dots(t\$t_1)\$\dots)\$t_n$ (with binary application operator \$), and $\lambda(x_1,\dots,x_n)$, μx and νx are regarded as unary operators.

$w \in \mathbb{N}^*$ is called an **occurrence of** $x \in Op$ **in** $\varphi \in \Lambda_\Sigma(V) \cup Fo_\Sigma(V)$ if $\varphi(w) = x$. $occ(x,\varphi)$ denotes the set of occurrences of x in φ. x **occurs in** φ if $occ(x,\varphi)$ is not empty.

$var(\varphi)$ denotes the set of $x \in V$ such that x occurs in φ.

An occurrence w of x is **bound in** φ if $\varphi(v)\{\lambda x, \forall x, \exists x\}$ for some prefix v of w. $bound(x,\varphi)$ denotes the set of bound occurrences of x in φ.

$x \in V$ is a **free variable** of φ if $occ(x,\varphi) \neq bound(x,\varphi)$. $free(\varphi)$ denotes the set of **free variables** of φ. φ is **closed** if $free(\varphi)$ is empty. φ is **monadic** if $free(\varphi)$ is a singleton.

A **Σ-substitution** is a $\mathcal{T}(S)$-sorted function $\sigma : V \to \Lambda_\Sigma(V)$.

σ is **finite** if the **support of** σ, $supp(\sigma) = \{x \in V \mid \sigma(x) \neq x\}$, is finite.

If σ is finite, it is also written as $\{\sigma(x)/x \mid x \in supp(\sigma)\}$.

Given $t : e \in T_\Sigma(V)$ (see section 9.3) and a Σ-substitution $\sigma : V \to \Lambda_\Sigma(V)$, then the **$\sigma$-instance of** t, $t\sigma : e \in \Lambda_\Sigma(V)$, is defined inductively as follows:

- For all $x \in V$, $x\sigma = \sigma(x)$,
- for all $c : \prod_{i\in I} e_i \to s \in F$ and $t \in \times_{i\in I} T_\Sigma(V)_{e_i}$, $c(t)\sigma = c(t\sigma)$,
- for all $e = \coprod_{i\in I} \prod_{j\in J} e_{ij} \in \mathcal{T}_s(S)$, $i \in I$ and $t \in \times_{j\in J} T_\Sigma(V)_{e_{ij}}$, $i(t)\sigma = \iota_i(t\sigma)$.

The **negation normal form** of a $\lambda\Sigma$-term $t : e$, $nf(t) : e$, is obtained by applying to t the following equations from left to right as often as possible:

$$\begin{aligned} \neg true &= false, \\ \neg false &= true, \\ \neg(t = t') &= t \neq t', \\ \neg(t \neq t') &= t = t', \end{aligned}$$

$$
\begin{aligned}
\neg\neg\varphi &= \varphi, \\
\neg(t \wedge t') &= \neg t \vee \neg t', \\
\neg(t \vee t') &= \neg t \wedge \neg t', \\
\neg(t \Rightarrow t') &= t \wedge \neg t', \\
\neg(t \ominus t') &= \neg t \vee t', \\
\neg(t \Leftarrow t') &= t \vee \neg t', \\
\neg(t \Leftrightarrow t') &= (t \wedge \neg t') \vee (\neg t \wedge t'), \\
\neg(t \oplus t') &= (t \wedge t') \vee (t \wedge t'), \\
\neg\overline{\forall}(r)(t) &= \overline{\exists}(r)(\neg t), \\
\neg\overline{\exists}(r)(t) &= \overline{\forall}(r)(\neg t), \\
\neg\nu x.t &= \mu x.\neg t[\neg x/x], \\
\neg\mu x.t &= \nu x.\neg t[\neg x/x].
\end{aligned}
$$

$t \in \Lambda_\Sigma(V)$ is **negation-free** if $nf(t)$ does not contain negation symbols.

The **negation normal form** of a Σ-formula φ, $nf(\varphi)$, is obtained by applying to φ the following equations from left to right as often as possible:

$$\begin{aligned}
\neg\, True &= False, \\
\neg False &= True, \\
\neg\neg\varphi &= \varphi, \\
\neg(\varphi \wedge \psi) &= \neg\varphi \vee \neg\psi, \\
\neg(\varphi \vee \psi) &= \neg\varphi \wedge \neg\psi, \\
\neg(\varphi \Rightarrow \psi) &= \varphi \wedge \neg\psi, \\
\neg(\varphi \ominus \psi) &= \neg\varphi \vee \psi, \\
\neg(\varphi \Leftarrow \psi) &= \varphi \vee \neg\psi, \\
\neg(\varphi \Leftrightarrow \psi) &= (\varphi \wedge \neg\psi) \vee (\neg\varphi \wedge \psi), \\
\neg(\varphi \oplus \psi) &= (\varphi \wedge \psi) \vee (\varphi \wedge \psi), \\
\neg\forall x\varphi &= \exists x\neg\varphi, \\
\neg\exists x\varphi &= \forall x\neg\varphi.
\end{aligned}$$

$\varphi \in Fo_\Sigma(V)$ is **negation-free up to** F if for all subformulas $\neg\psi$ of φ, ψ is an (S,F)-formula.

Let $\varphi \in \Lambda_\Sigma(V) \cup Fo_\Sigma(V)$. In terms of the above tree representation, $w \in def(nf(\varphi))$ has **positive** (**negative**) **polarity** if the number of prefixes v of w with $\varphi(v) = \neg$ is even (odd).

10.3 Semantics

Let Σ be as above, $\mathcal{A}$ be a Σ-algebra with carrier A, let

$$Q = A_{state},\ L = A_{label} \text{ and } At = A_{atom}.$$

$\mathcal{A}$ is **finitely branching** if for all $q \in Q$ and $lab \in L$, $trans^{\mathcal{A}}(q)$ and $trans^{\mathcal{A}}(q)(lab)$ are finite.

Let V be a $\mathcal{T}(S)$-sorted set of variables. A **valuation of V in A** is an $\mathcal{T}(S)$-sorted function from V to A, i.e., a function tuple $g = (g_e : V_e \to A_e)_{e \in \mathcal{T}(S)}$ (see chapter 7). The set of valuations of V in A is denoted by A^V.

Valuations may be regarded as the states of (many-sorted) predicate logic. Besides S-sorted variables, V includes variables for unary and binary state relations ($\mathcal{P}(state)$ and $\mathcal{P}(state^2)$), respectively), mainly in order to represent relational fixpoints as Σ-formulas.

Suppose that $\mathcal{A}$ is finitely branching.

For all $t : e \in \Lambda_\Sigma(V)$, the **interpretation of t in $\mathcal{A}$**, $t^{\mathcal{A}} : A^V \to A_e$, is inductively defined: Let $g \in A^V$. If t is closed, we often abbreviate $t^{\mathcal{A}}(g)$ to $t^{\mathcal{A}}$.

- For all $I \subseteq \mathcal{I}$ and $i \in I$, $i^{\mathcal{A}}(g) = i$.

- For all $x \in V$, $x^{\mathcal{A}}(g) = g(x)$.
- For all $f \in Arr_\Sigma$, $f^{\mathcal{A}}$ is defined as in section 9.1.
- For all $t \in T_\Sigma(V)$, $t^{\mathcal{A}}(g) = g^*(t)$ (see section 9.11).
- For all $I \subseteq \mathcal{I}$, $(f_i : e_i \to e)_{i \in I} \in \Lambda_\Sigma(V)^I$, $i \in I$ and $a \in A_{e_i}$, $[f_i]_{i\in I}^{\mathcal{A}}(a) = f_i^{\mathcal{A}}(a)$.
- For all $I \subseteq \mathcal{I}$, $i \in I$ and $a \in A_{e_i}$, $\iota_i^{\mathcal{A}}(a) = a$.
- For all $e \in \mathcal{T}(S)$, $(true : \mathcal{P}(e))^{\mathcal{A}} = A_e$.
- For all $b, c \in 2$, $\neg^{\mathcal{A}}(b) = 1 - b$ and $b \wedge^{\mathcal{A}} c = b * c$.
- For all $\varphi, \psi \subseteq A_e$, $\neg^{\mathcal{A}}(\varphi) = A_e \setminus \varphi$ and $\varphi \wedge^{\mathcal{A}} \psi = \varphi \cap \psi$.
- For all $a \in 2$ and $b, c \in A_e$, $ite^{\mathcal{A}}(a, b, c) = \begin{cases} b & \text{if } a = 1, \\ c & \text{if } a = 0. \end{cases}$
- For all $e \in \mathcal{T}(S)$, $(=)^{\mathcal{A}} = \{(a, a) \mid a \in A_e\}$.
- For all $(t_i : e_i)_{i \in I} \in \Lambda_\Sigma(V)^I$, $(t_i)_{i\in I}^{\mathcal{A}}(g) = (t_i^{\mathcal{A}}(g))_{i \in I}$.
- For all $t : e,\ f : e \to e' \in \Lambda_\Sigma(V)$, $f(t)^{\mathcal{A}}(g) = f^{\mathcal{A}}(g)(t^{\mathcal{A}}(g))$.
- For all $x : e \in V$, $t : e' \in \Lambda_\Sigma(V)$ and $a \in A_e$,

$$(\lambda x.t)^{\mathcal{A}}(g)(a) = t^{\mathcal{A}}(g[a/x])$$

(see chapter 2).

- For all $a \in A_e$, $single^{\mathcal{A}}(a) = \{a\}$.
- For all $\varphi \subseteq A_e$, $card^{\mathcal{A}}(\varphi) = |\varphi|$.
- For all $f : A_e \to A_{e'}$, $map^{\mathcal{A}}(f) = \mathcal{P}(f)$ (see section 2.8).
- For all $p : A_e \to 2$ and $\varphi \subseteq A_e$, $filter^{\mathcal{A}}(p)(\varphi) = \{a \in \varphi \mid p(a) = 1\}$.
- For all $e \in \mathcal{T}(S)$ and $\varphi \subseteq A_e$, $join^{\mathcal{A}}(\varphi) = \bigcup \varphi$ and $meet^{\mathcal{A}}(B) = \bigcap B$.
- For all $f : A_e \to A_{e'} \to A_e$, $a \in A_e$, $b \in A_{e'}$ and $w \in A_{e'*}$,

$$\begin{aligned} foldl^{\mathcal{A}}(f)(a)(\epsilon) &= a, \\ foldl^{\mathcal{A}}(f)(a)(b \cdot w) &= foldl^{\mathcal{A}}(f)(f(a)(b))(w). \end{aligned}$$

- For all $f : A_e \to \mathcal{P}(A_e)$, $f' : A_e \to A_{e'} \to \mathcal{P}(A_e)$, $a \in A_e$, $b \in A_{e'}$ and $w \in A_{e'*}$,

$$\begin{aligned} foldNDS^{\mathcal{A}}(f)(f')(a)(\epsilon) &= f(a), \\ foldNDS^{\mathcal{A}}(f)(f')(a)(b \cdot w) &= f(a) >\!\!>\!= \lambda x.f'(x)(b) \\ &\quad >\!\!>\!= \lambda x.foldNDS^{\mathcal{A}}(f)(f')(x)(w). \end{aligned}$$

- For all $\varphi \subseteq A_e$ and $a \in A_e$, $\chi^{\mathcal{A}}(\varphi)(a) = 1 \Leftrightarrow_{def} a \in \varphi$.
- For all $r \subseteq A_e \times A_{e'}$ and $a \in A_e$, $rel2fun^{\mathcal{A}}(r)(a) = \{b \in A_{e'} \mid (a, b) \in r\}$.

- For all $\varphi \subseteq A_e$, $\psi \subseteq A_{e'}$, $r \subseteq A_e \times A_{e'}$ and $r' \subseteq A'_{e'} \times A_{e''}$,

$$\begin{aligned}
inv^{\mathcal{A}}(r) &= \{(b,a) \mid (a,b) \in r\},\\
\pi_1^{\mathcal{A}}(r) &= \{a \in A_e \mid \exists\, b \in A_{e'} : (a,b) \in r\},\\
\pi_2^{\mathcal{A}}(r) &= \{b \in A_{e'} \mid \exists\, a \in A_e : (a,b) \in r\},\\
\varphi *^{\mathcal{A}} \psi &= \varphi \times \psi,\\
r/^{\mathcal{A}}\psi &= \{a \in A_e \mid \forall\, b \in \psi : (a,b) \in r\},\\
\overline{\forall}^{\mathcal{A}}(r)(\psi) &= \{a \in A_e \mid \forall\, b \in A_{e'} : (a,b) \in r \Rightarrow b \in \psi\},\\
r;^{\mathcal{A}} r' &= \{(a,c) \in A_e \times A_{e''} \mid \exists\, b \in A_{e'} : (a,b) \in r \wedge (b,c) \in r'\}.
\end{aligned}$$

- For all $x : \mathcal{P}(e) \in V$ and negation-free $\varphi : \mathcal{P}(e) \in \Lambda_\Sigma(V)$,

$$(\mu x.\varphi)^{\mathcal{A}}(g) \;=\; \bigcap\{B \subseteq A_e \mid \varphi^{\mathcal{A}}(g[B/x]) \subseteq B\}.$$

  By Theorem 3.9 (1) and Proposition 10.5, $(\mu x.\varphi)^{\mathcal{A}}(g)$ is the least fixpoint of the step function

$$\lambda B.\varphi^{\mathcal{A}}(g[B/x]) : \mathcal{P}(A_e) \to \mathcal{P}(A_e).$$

- For all $x : e \in V$ and $\varphi : e,\ \psi : e' \in \Lambda_\Sigma(V)$,

$$\psi[\varphi/x]^{\mathcal{A}}(g) = \psi^{\mathcal{A}}(g[\varphi^{\mathcal{A}}(g)/x]).$$

- $\sim^{\mathcal{A}} = \Phi_\infty = \bigcap_{i\in\mathbb{N}} \Phi^i(Q^2) \subseteq \mathcal{P}(Q^2)$ where $\Phi : \mathcal{P}(Q^2) \to \mathcal{P}(Q^2)$ maps $R_{state} \subseteq Q^2$ to

$$\{(q,q') \in Q^2 \mid out^{\mathcal{A}}(q) = out^{\mathcal{A}}(q'),\ (trans^{\mathcal{A}}(q), trans^{\mathcal{A}}(q')) \in R_{\mathcal{P}(state)}\}$$
$$\cap \bigcap_{lab\in L}\{(q,q') \in Q^2 \mid \left\{ \begin{array}{l} outL^{\mathcal{A}}(q,lab) = outL^{\mathcal{A}}(q',lab), \\ (trans^{\mathcal{A}}(q,lab), trans^{\mathcal{A}}(L')(q',lab)) \in R_{\mathcal{P}(state)} \end{array} \right\}\}$$

  and $R_{\mathcal{P}(state)} \subseteq \mathcal{P}(Q) \times \mathcal{P}(Q)$ is the lifting of R_{state} according to section 7.2.

  Since Φ is monotone and, by Theorem 10.7, $\sim^{\mathcal{A}}$ is Φ-dense, Theorem 3.4 (2) implies that $\sim^{\mathcal{A}}$ is the greatest fixpoint of Φ.
- $labels^{\mathcal{A}} = L$.
- For all $q \in Q$, $preds^{\mathcal{A}}(q) = \{q' \in Q \mid q \in trans^{\mathcal{A}}(q')\}$.
- For all $q \in Q$ and $lab \in L$, $predsL^{\mathcal{A}}(q)(lab) = \{q' \in Q \mid q \in transL^{\mathcal{A}}(q')(lab)\}$.
- For all $q \in Q$, $out^{\mathcal{A}}(q) = \{at \in At \mid q \in value^{\mathcal{A}}(at)\}$.
- For all $q \in Q$ and $lab \in L$, $outL^{\mathcal{A}}(q,lab) = \{at \in At \mid q \in valueL^{\mathcal{A}}(at)(lab)\}$.
- For all $L' \subseteq L$,

$$child^{\mathcal{A}}(L') = \begin{cases} \{(q,q') \in Q^2 \mid q' \in trans^{\mathcal{A}}(q)\} & \text{if } L' = \emptyset, \\ \{(q,q') \in Q^2 \mid q' \in \bigcup_{lab\in L'} transL^{\mathcal{A}}(q)(lab)\} & \text{otherwise.} \end{cases}$$

- For all $q_0, q \in Q$,

$$
\begin{aligned}
&traces^{\mathcal{A}}(q)(q_{fin}) = f\{q\}(q) \text{ where} \\
&\quad f(visited)(q) = (trans^{\mathcal{A}}(q) \\
&\qquad\qquad >\!\!>\!= \lambda q.if\ \ q = q_{fin}\ then\ \{next\} \\
&\qquad\qquad\qquad else\ if\ \ q \in visited\ then\ \emptyset \\
&\qquad\qquad\qquad else\ f(visited \cup \{q\})(q) >\!\!>\!= \lambda trace.\{next \cdot trace\} \\
&\qquad\qquad\qquad \text{where } next = \iota_1(q)) \cup \\
&\qquad\quad (L >\!\!>\!= \lambda lab.transL^{\mathcal{A}}(q)(lab) \\
&\qquad\qquad >\!\!>\!= \lambda q.if\ \ q = q_{fin}\ then\ \{next\} \\
&\qquad\qquad\qquad else\ if\ \ q \in visited\ then\ \emptyset \\
&\qquad\qquad\qquad else\ f(visited \cup \{q\})(q) >\!\!>\!= \lambda trace.\{next \cdot trace\} \\
&\qquad\qquad\qquad \text{where } next = \iota_2(lab, q)).
\end{aligned}
$$

- For all $q \in Q$ and $lab \in L$,

$$transL2row^{\mathcal{A}}(q)(lab) = \begin{cases} q' & \text{if } \exists\, q' \in Q : transL(q)(lab) = \{q'\}, \\ () & \text{otherwise.} \end{cases}$$

- For all $val, val' \in A_{row}$ and $lab \in L$,

$$(val +^{\mathcal{A}} val')(lab) = \textit{if } val(lab) \neq () \textit{ then } val(lab) \textit{ else } val'(lab).$$

- For all $L' \subseteq L$, $val \in A_{row}$ and $lab \in L'$,

$$projectrow^{\mathcal{A}}(L')(val)(lab) = \textit{if } lab \in L' \textit{ then } val(lab) \textit{ else } ().$$

- For all $f : L \to Q$ and $val \in A_{row}$,

$$selectrow^{\mathcal{A}}(f)(val) = \begin{cases} 1 & \text{if } \forall\, lab \in L : val(lab) \in \{(), f(lab)\}, \\ 0 & \text{otherwise.} \end{cases}$$

- For all $tab, tab' \in A_{table}$,

$$
\begin{aligned}
labs^{\mathcal{A}}(tab) &= \{lab \in L \mid \exists\ val \in tab : val(lab) \neq ()\},\\
tab \wedge^{\mathcal{A}} tab' &= tab \cap tab',\\
tab *^{\mathcal{A}} tab' &= if\ \ labs(tab) \cap labs(tab') = \emptyset\ then\ tab \times tab'\ else\ \emptyset,\\
tab\ /^{\mathcal{A}}\ tab' &= if\ \ labs(tab') \subseteq labs(tab) \wedge tab' \neq \emptyset\\
&\quad then\ \{\pi(row) \mid row \in tab,\\
&\qquad\qquad \forall\ row' \in tab' : \pi(row) + row' \in tab\}\\
&\quad else\ \emptyset\\
&\quad \text{where } \pi = projectrow^{\mathcal{A}}(labs(tab) \setminus labs(tab')),\\
njoin^{\mathcal{A}}(tab)(tab') &= filter(equal)(tab \times tab')\\
&\quad \text{where for all } val \in tab \text{ and } val' \in tab',\\
&\quad equal(val, val') = \begin{cases} 1 & \text{if } \forall\ lab \in L : val(lab) = val'(lab) \vee\\ & \qquad\qquad () \in \{val(lab), val'(lab)\},\\ 0 & \text{otherwise.} \end{cases}
\end{aligned}
$$

- For all $tab \in A_{table} = \mathcal{P}(A_{row})$, $lab \in L$ and $P \subseteq \mathcal{P}(L)$,

$$fundeps^{\mathcal{A}}(tab) = \bigcup \mathcal{P}(mindeps)(labs(tab)) \text{ where}$$

$$mindeps(lab) = \{(lab, L') \in deps \mid L' \in minis(rel2fun(deps)(lab))\} \text{ where}$$

$$deps = \{(lab, L') \in L \times \mathcal{P}(L)$$

$$\mid lab \notin L' \neq \emptyset,\ \forall\ vals \in \pi_1(rel) : |rel2fun(rel)(vals)| = 1\}$$

$$\text{where } rel = map(\lambda val.(map(val)(L'), val(lab)))(tab),$$

$$minis(P) = \{L' \in P \mid \forall\ L'' \in P \setminus \{L'\} : L'' \not\subseteq L'\}.$$

Consequently, for all $(lab, L') \in deps$ and $val, val' \in tab$,

$$map(val)(L') = map(val')(L') \quad \Rightarrow \quad val(lab) = val'(lab).$$

For all substitutions $\sigma : V \to \Lambda_\Sigma(V)$, the **interpretation of σ in $\mathcal{A}$**, $\sigma^{\mathcal{A}} : A^V \to A^V$, is defined as folllows:

For all $g \in A^V$ and $x \in V$, $\sigma^{\mathcal{A}}(g)(x) = \sigma(x)^{\mathcal{A}}(g)$.

Proposition 10.1

For all $\mathcal{A} \in Alg_\Sigma$ with carrier A, $t \in T_\Sigma(V)$ and Σ-substitutions $\sigma : V \to \Lambda_\Sigma(V)$,

$$(t\sigma)^{\mathcal{A}} = t^{\mathcal{A}} \circ \sigma^{\mathcal{A}}$$

(see section 9.11).

Proof by induction on t. Let $g \in A^V$

For all $x \in V$, $(x\sigma)^{\mathcal{A}}(g) = (\sigma(x))^{\mathcal{A}}(g) = \sigma^{\mathcal{A}}(g)(x) = x^{\mathcal{A}}(\sigma^{\mathcal{A}}(g))$.

For all $c : \prod_{i \in I} e_i \to s \in F$ and $t \in \times_{i \in I} T_\Sigma(V)_{e_i}$,

$$(c(t)\sigma)^{\mathcal{A}}(g) = (c(t\sigma))^{\mathcal{A}}(g) = c^{\mathcal{A}}((t\sigma)^{\mathcal{A}}(g)) \overset{ind.\ hyp.}{=} c^{\mathcal{A}}(t^{\mathcal{A}}(\sigma^{\mathcal{A}}(g))) = c(t)^{\mathcal{A}}(\sigma^{\mathcal{A}}(g)).$$

For all $e = \coprod_{i \in I} \prod_{j \in J} e_{ij} \in \mathcal{T}_s(S)$, $i \in I$ and $t \in \times_{j \in J} T_\Sigma(V)_{e_{ij}}$,

$$(i(t)\sigma)^{\mathcal{A}}(g) = (\iota_i(t\sigma))^{\mathcal{A}}(g) = \iota_i((t\sigma)^{\mathcal{A}}(g)) \overset{ind.\ hyp.}{=} \iota_i(t^{\mathcal{A}}(\sigma^{\mathcal{A}}(g))) = \iota_i(t)^{\mathcal{A}}(\sigma^{\mathcal{A}}(g)).$$

❑

Lemma 10.2

Σ-terms can be turned into Σ-arrows such that Σ-homomorphisms remain Arr_Σ-homomorphic (see Lemma 9.2).

Proof. Let $e \in \mathcal{T}_{po}(S)$, $t \in T_\Sigma(V)_e$ and $var(t) = \{x_1 : s_1, \ldots, x_n : s_n\}$. Interpret the Σ-arrow

$$\lambda(x_1, \ldots, x_n).t : \prod_{i=1}^{n} s_i \to e \tag{1}$$

in a Σ-algebra $\mathcal{A}$ with carrier A as follows:

Let $e' = \prod_{i=1}^{n} s_i$. For all $a \in A_e$,

$$(\lambda(x_1, \ldots, x_n).t)^{\mathcal{A}}(a) = g_a^*(t)$$

where $g_a \in A^V$ is defined by $g_a(x_i) = \pi_i(a)$ for all $1 \leq i \leq n$.

Then for all Σ-homomorphisms $h : \mathcal{A} \to \mathcal{B}$ and $x \in V$,

$$h(g_a(x)) = h(\pi_x(a)) = \pi_x(h(a)) = g_{h(a)}(x)$$

and thus

$$h((\lambda(x_1, \ldots, x_n).t)^{\mathcal{A}}(a)) = h(g_a^*(t)) \stackrel{Lemma\ 9.9}{=} (h \circ g_a)^*(t) = g_{h(a)}^*(t)$$
$$= (\lambda(x_1, \ldots, x_n).t)^{\mathcal{B}}(h(a)).$$

❑

For all $\varphi \in Fo_\Sigma(V)$, the **interpretation of φ in $\mathcal{A}$**, $\varphi^\mathcal{A} \subseteq A^V$, is inductively defined as follows:

- $True^\mathcal{A} = A^V$.
- For all $p : \mathcal{P}(e),\ t : e \in \Lambda_\Sigma(V)$, $p(t)^\mathcal{A} = \{g \in A^V \mid t^\mathcal{A}(g) \in p^\mathcal{A}(g)\}$.
- For all $\varphi : Fo_\Sigma(V)$, $(\neg\varphi)^\mathcal{A} = A^V \setminus \varphi^\mathcal{A}$.
- For all $\varphi, \psi : Fo_\Sigma(V)$, $(\varphi \wedge \psi)^\mathcal{A} = \varphi^\mathcal{A} \cap \psi^\mathcal{A}$.
- For all $e \in \mathcal{T}(S)$, $x : e \in V$ and $\varphi \in Fo_\Sigma(V)$,

$$(\forall x\varphi)^\mathcal{A} = \bigcap_{a \in A_e} \{g \in A^V \mid g[a/x] \in \varphi^\mathcal{A}\}.$$

$g \in A^V$ **solves** $\varphi \in Fo_\Sigma(V)$ in $\mathcal{A}$ if $g \in \varphi^\mathcal{A}$.

Hence the elements of $\varphi^\mathcal{A}$ are the **solutions** of φ.

$g \in A^V$ **solves** $\varphi \in Fo_\Sigma(V)$ **uniquely** in $\mathcal{A}$ if $g \in \varphi^\mathcal{A}$ and for all $h \in \varphi^\mathcal{A}$,

$$h|_{free(\varphi)} = g|_{free(\varphi)}$$

(see chapter 2 and section 10.2).

A $\mathcal{T}(S)$-sorted function $f : V \to \Lambda_\Sigma(V)$ **solves** $\varphi \in Fo_\Sigma(V)$ **uniquely** in $\mathcal{A}$ if f solves φ in $\mathcal{A}$ and for all solutions f' of φ in $\mathcal{A}$ and $g \in A^V$,

$$\lambda x.f(x)^{\mathcal{A}}(g)|_{free(\varphi)} = \lambda x.f'(x)^{\mathcal{A}}(g)|_{free(\varphi)}.$$

These definitions are motivated by the following result:

Proposition 10.3

Let $g, h \in A^V$, $t \in \Lambda_\Sigma(V)$ and $\varphi \in Fo_\Sigma(V)$.
If $g|_{free(t)} = h|_{free(t)}$, then $t^{\mathcal{A}}(g) = t^{\mathcal{A}}(h)$.
If $g|_{free(\varphi)} = h|_{free(\varphi)}$, then g solves φ in $\mathcal{A}$ iff h solves φ in $\mathcal{A}$. ❏

Due to Proposition 10.3, we omit the valuation parameter of $t^{\mathcal{A}}$ whenever $free(t)$ is empty.

A $\mathcal{T}(S)$-sorted function $f : V \to \Lambda_\Sigma(V)$ **solves** $\varphi \in Fo_\Sigma(V)$ in $\mathcal{A}$ if for all $g \in A^V$,

$$\lambda x.f(x)^\mathcal{A}(g) \in \varphi^\mathcal{A}.$$

$\mathcal{A}$ **satisfies** $\varphi \in Fo_\Sigma(V)$, written as $\mathcal{A} \models \varphi$, if $\varphi^\mathcal{A} = A^V$.

$\mathcal{A}$ satisfies $\varphi \Rightarrow \psi \in Fo_\Sigma(V)$ iff $\varphi^\mathcal{A} \subseteq \psi^\mathcal{A}$.

$\mathcal{A}$ **satisfies** $AX \subseteq Fo_\Sigma(V)$, written as $\mathcal{A} \models AX$, if $\mathcal{A}$ satisfies all elements of AX.

A class $\mathcal{K}$ of Σ-algebras **satisfies** AX if all $\mathcal{A} \in \mathcal{K}$ satisfy AX.

Two λΣ-terms or Σ-formulas φ and ψ are **$\mathcal{A}$-equivalent** if $\varphi^\mathcal{A} = \psi^\mathcal{A}$.

Proposition 10.4 λΣ-terms and Σ-formulas are $\mathcal{A}$-equivalent to their respective negation normal forms (see section 10.2). ❏

Proposition 10.5 For all negation-free λΣ-terms $\varphi : \mathcal{P}(e)$, $g \in A^V$ and $x : \mathcal{P}(e) \in V$, the function

$$\lambda S.\varphi^\mathcal{A}(g[S/x]) : \mathcal{P}(A_e) \to \mathcal{P}(A_e)$$

is monotone w.r.t. the subset relation.

Proof. By Proposition 10.4, it is sufficient to show that for all $\varphi, \varphi', \psi, \psi' : \mathcal{P}(e)$, $r : \mathcal{P}(e \times e') \in Fo_\Sigma(V)$, $x : \mathcal{P}(e) \in V$ and $g \in A^V$, $\varphi^{\mathcal{A}}(g) \subseteq \varphi'^{\mathcal{A}}(g) \wedge \psi^{\mathcal{A}}(g) \subseteq \psi'^{\mathcal{A}}(g)$ implies

$$(\varphi \wedge \psi)^{\mathcal{A}}(g) \subseteq (\varphi' \wedge \psi')^{\mathcal{A}}(g), \quad (\varphi \vee \psi)^{\mathcal{A}} \subseteq (\varphi' \vee \psi')^{\mathcal{A}}(g), \tag{1}$$

$$\overline{\exists}(r)(\varphi)^{\mathcal{A}} \subseteq \overline{\exists}(r)(\varphi')^{\mathcal{A}}(g), \quad \overline{\forall}(r)(\varphi)^{\mathcal{A}} \subseteq \overline{\forall}(r)(\varphi')^{\mathcal{A}}(g), \tag{2}$$

$$(\mu x.\varphi)^{\mathcal{A}} \subseteq (\mu x.\varphi')^{\mathcal{A}}(g), \quad (\nu x.\varphi)^{\mathcal{A}} \subseteq (\nu x.\varphi')^{\mathcal{A}}(g). \tag{3}$$

The proof of (1)-(3) is left to the reader. ❑

Proposition 10.6 For all $r \subseteq A_e \times A_{e'}$, $\psi \subseteq A_{e'}$, $g \in A^V$ and $tab, tab' \in A_{table}$,

$$\begin{aligned}
(r/\psi)^{\mathcal{A}} &= \pi_1(r) \ominus \pi_1((\pi_1(r) * \psi) \ominus r)^{\mathcal{A}}, \\
((r/\psi) * \psi)^{\mathcal{A}} &\subseteq r^{\mathcal{A}}, \\
tab \;/^{\mathcal{A}}\; tab' &= \mathit{if}\ labs(tab') \subseteq labs(tab) \wedge tab' \neq \emptyset \\
&\quad\ \mathit{then}\ \pi(tab) \setminus \pi((\pi(tab) \times tab') \setminus tab)\ \mathit{else}\ \emptyset \\
&\quad\ \text{where } \pi = project^{\mathcal{A}}(labs(tab) \setminus labs(tab')).
\end{aligned}$$

❑

The above semantics of universal quantifiers and μ-abstraction and the derivation of existential quantifiers and ν-abstraction in section 10.2 reveals that $\exists$, $\overline{\exists}$ and ν are interpreted dually to $\forall$, $\overline{\forall}$ and μ, respectively:

For all $x:e \in V$ and $\varphi \in Fo_\Sigma(V)$,

$$(\exists x\varphi)^{\mathcal{A}} = (\neg\forall x\neg\varphi)^{\mathcal{A}} = A^V \setminus \bigcap_{a\in A_e}\{g \in A^V \mid g[a/x] \in (\neg\varphi)^{\mathcal{A}}\}$$
$$= \bigcup_{a\in A_e}(A^V \setminus \{g \in A^V \mid g[a/x] \in A^V \setminus \varphi^{\mathcal{A}}\}) = \bigcup_{a\in A_e}\{g \in A^V \mid g[a/x] \notin A^V \setminus \varphi^{\mathcal{A}}\}$$
$$= \bigcup_{a\in A_e}\{g \in A^V \mid g[a/x] \in \varphi^{\mathcal{A}}\},$$

For all $r:\mathcal{P}(e\times e')$, $\psi:\mathcal{P}(e') \in \Lambda_\Sigma(V)$ and $g \in A^V$,

$$\overline{\exists}(r)(\psi)^{\mathcal{A}}(g) = (\neg\overline{\forall}(r)(\neg\psi))^{\mathcal{A}}(g) = A_e \setminus \{a \in A_e \mid \mathit{rel2fun}(r)^{\mathcal{A}}(g)(a) \subseteq (\neg\psi)^{\mathcal{A}}\}$$
$$= \{a \in A_e \mid \mathit{rel2fun}(r)^{\mathcal{A}}(g)(a) \not\subseteq A_{e'} \setminus \psi^{\mathcal{A}}\} = \{a \in A_e \mid \mathit{rel2fun}(r)^{\mathcal{A}}(g)(a) \cap \psi^{\mathcal{A}} \neq \emptyset\}.$$

For all $x : \mathcal{P}(e) \in V$, negation-free $\varphi : \mathcal{P}(e) \in Fo_\Sigma(V)$ and $g \in A^V$,

$$
\begin{aligned}
&(\nu x.\varphi)^{\mathcal{A}}(g) = (\neg\mu x.\neg\varphi[\neg x/x])^{\mathcal{A}}(g) = A_e \setminus (\mu x.\neg\varphi[\neg x/x])^{\mathcal{A}}(g)\\
&= A_e \setminus \bigcap\{B \subseteq A_e \mid (\neg\varphi[\neg x/x])^{\mathcal{A}}(g[B/x]) \subseteq B\}\\
&= A_e \setminus \bigcap\{B \subseteq A_e \mid A_e \setminus \varphi[\neg x/x]^{\mathcal{A}}(g[B/x]) \subseteq B\}\\
&= A_e \setminus \bigcap\{B \subseteq A_e \mid A_e \setminus \varphi^{\mathcal{A}}(g[B/x][(\neg x)^{\mathcal{A}}(g[B/x])/x]) \subseteq B\}\\
&= A_e \setminus \bigcap\{B \subseteq A_e \mid A_e \setminus \varphi^{\mathcal{A}}(g[(\neg x)^{\mathcal{A}}(g[B/x])/x]) \subseteq B\\
&= A_e \setminus \bigcap\{B \subseteq A_e \mid A_e \setminus \varphi^{\mathcal{A}}(g[(A_e \setminus x^{\mathcal{A}}(g[B/x]))/x]) \subseteq B\}\\
&= A_e \setminus \bigcap\{B \subseteq A_e \mid A_e \setminus \varphi^{\mathcal{A}}(g[(A_e \setminus g[B/x](x))/x]) \subseteq B\}\\
&= A_e \setminus \bigcap\{B \subseteq A_e \mid A_e \setminus \varphi^{\mathcal{A}}(g[(A_e \setminus B)/x]) \subseteq B\}\\
&= \bigcup\{A_e \setminus B \mid B \subseteq A_e,\ A_e \setminus \varphi^{\mathcal{A}}(g[(A_e \setminus B)/x]) \subseteq B\}\\
&= \bigcup\{A_e \setminus B \mid B \subseteq A_e,\ A_e \setminus B \subseteq \varphi^{\mathcal{A}}(g[(A_e \setminus B)/x])\}\\
&= \bigcup\{B \subseteq A_e \mid B \subseteq \varphi^{\mathcal{A}}(g[B/x])\}.
\end{aligned}
$$

Hence by Theorem 3.9 (5) and Proposition 10.5, $(\nu x.\varphi)^{\mathcal{A}}(g)$ is the greatest fixpoint of the step function

$$\lambda B.\varphi^{\mathcal{A}}(g[B/x]) : \mathcal{P}(A_e) \to \mathcal{P}(A_e).$$

Theorem 10.7 Suppose that $\mathcal{A}$ is finitely branching. $\Phi_\infty \subseteq \mathcal{P}(Q^2)$ with Φ as defined above is Φ-dense.

Proof. Let $(q,q') \in \Phi_\infty$. Then

$$\text{for all } i \in \mathbb{N},\ (q,q') \in \Phi^i(Q^2). \tag{1}$$

We must show $(q,q') \in \Phi(\Phi_\infty)$, i.e.,

$$out^{\mathcal{A}}(q) = out^{\mathcal{A}}(q'), \tag{2}$$
$$\forall\ lab \in L : outL^{\mathcal{A}}(q, lab) = outL^{\mathcal{A}}(q', lab), \tag{3}$$
$$\forall\ lab \in L : (trans^{\mathcal{A}}(q, lab), trans^{\mathcal{A}}(q', lab)) \in R_{\mathcal{P}(state)} \tag{4}$$

where $R_{\mathcal{P}(state)}$ is the lifting of $R_{state} = \Phi_\infty$ according to section 7.2.

(2) and (3) follow from $(q, q') \in \Phi(Q^2)$.

Let $r \in trans^{\mathcal{A}}(q)$. By (1), for all $i \in \mathbb{N}$ there is $r_i \in trans^{\mathcal{A}}(q')$ with $(r, r_i) \in \Phi^i(Q^2)$. Since $\mathcal{A}$ be finitely branching, $trans^{\mathcal{A}}(q)$ is finite and thus there is $r' \in Q$ with $r' = r_n \in trans^{\mathcal{A}}(q')$ for infinitely many $n \in \mathbb{N}$. Hence for all $n \in \mathbb{N}$ there is $i_n \geq n$ with $r_{i_n} = r'$. Consequently, $(r, r_{i_n}) \in \Phi^{i_n}(Q^2)$ implies $(r, r') \in \Phi^{i_n}(Q^2) \subseteq \Phi^n(Q^2)$. Therefore, $(r, r') \in \bigcap_{n \in \mathbb{N}} \Phi^n(Q^2) = \Phi_\infty$.

Analogously, for all $r' \in trans^{\mathcal{A}}(q')$ there is $r \in trans^{\mathcal{A}}(q)$ with $(r, r') \in \Phi_\infty$. Hence we conclude (4).

(5) can be shown analogously. ❏

Theorem 10.8 (fixpoint induction and coinduction for set types)

Let $\varphi : \mathcal{P}(e) \in Fo_\Sigma(V)$ be negation-free, $free(\varphi) = \{x : \mathcal{P}(e)\}$, $g \in A^V$,

$$\begin{array}{rcl} F : \mathcal{P}(A_e) & \to & \mathcal{P}(A_e) \\ B & \mapsto & \varphi^{\mathcal{A}}(g[B/x]), \end{array}$$

$B \subseteq A_e$, $R \subseteq Q^2$ and Φ be as in the above definition of $\sim^{\mathcal{A}}$. (Proposition 10.3 ensures that $F(B)$ is unique.)

(1) If B is F-closed, then $(\mu x.\varphi)^{\mathcal{A}} \subseteq B$.

(2) If B is F-dense, then $B \subseteq (\nu x.\varphi)^{\mathcal{A}}$.

(3) If R is Φ-dense, then $R \subseteq \sim^{\mathcal{A}}$.

Proof. By Proposition 10.5, F is monotone. Hence Theorem 3.13 (1) implies (1) and Theorem 3.14 (1) implies (2) and (3). ❏

Examples

Every natural number is even or odd: Let $\Sigma = Nat$ (see section 8.2), $e = nat$, $\varphi = single(zero) \vee map(succ)(x)$ (see section 10.1), $\mathcal{A}$ be the Nat-algebra of 9.6.1 and

$$F = \lambda B.\{0\} \cup \{n+1 \mid n \in B\} : \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N}).$$

In section 3.2, we have shown that $\mathbb{N}$ is the least fixpoint of F, i.e., $\mathbb{N} = (\mu x.\varphi)^{\mathcal{A}}$. Hence by Theorem 10.8 (1), it is sufficient to show that the set B of even or odd natural numbers is F-closed. So let $n \in F(B)$. Then $n = 0$ or $n = m+1$ for some even or odd natural number m.

If $n = 0$, then n is even. If $n = m+1$ and m is even, then n is odd. If $n = m+1$ and m is odd, then n is even. Hence $n \in B$ in all three cases. Therefore, B is F-closed. ❏

blink has infinitely many zeros: Let $\Sigma = \mathit{Stream}(\mathbb{N})$ (see section 8.3), $e = \mathit{state}$,

$$\varphi = \mathit{filter}(\lambda s.(\mathit{head}(s) = 0 \vee (\chi(x) \circ \mathit{tail})(s)))(\mathit{true}),$$
$$\psi = \mathit{filter}(\chi(x) \circ \mathit{tail})(\mu x.\varphi)$$

(see section 10.1), $\mathcal{A}$ be the $\mathit{Stream}(\mathbb{N})$-algebra $\mathit{InfSeq}(\mathbb{N})$ (see 9.6.5) and

$$F = \lambda B.\{s \in \mathbb{R}^{\mathbb{N}} \mid \mathit{head}(s) = 0 \vee \mathit{tail}(s) \in B\} : \mathcal{P}(\mathbb{R}^{\mathbb{N}}) \to \mathcal{P}(\mathbb{R}^{\mathbb{N}}),$$
$$G = \lambda B.\{s \in \mathit{has}0 \mid \mathit{tail}(s) \in B\} : \mathcal{P}(\mathbb{R}^{\mathbb{N}}) \to \mathcal{P}(\mathbb{R}^{\mathbb{N}}).$$

In section 3.2, we have shown that $\mathit{has}0$ is the least fixpoint of F, i.e., $\mathit{has}0 = (\mu x.\varphi)^{\mathcal{A}}$ and $\mathit{has}\infty 0$ is the greatest fixpoint of G, i.e., $\mathit{has}0\infty 0 = (\nu x.\psi)^{\mathcal{A}}$.

Hence by Theorem 10.8 (2), it is sufficient to show that

$$\mathit{blink} = \lambda n.\mathit{if}\ n\ \mathit{is\ even\ then}\ 0\ \mathit{else}\ 1$$

belongs to a G-dense subset B of $\mathbb{R}^{\mathbb{N}}$. Let $B = \{\mathit{blink}, \lambda n.\mathit{blink}(n+1)\}$. Then

$$B \subseteq \{s \in \mathit{has}0 \mid \mathit{tail}(s) \in B\} = G(B),$$

i.e., B is G-dense. ❏

10.4 Realization in Expander2

The quintuple $(inits^{\mathcal{A}}, trans^{\mathcal{A}}, transL^{\mathcal{A}}, value^{\mathcal{A}}, valueL^{\mathcal{A}})$ is called the **Kripke model of $\mathcal{A}$**.

Expander2 generates the Kripke model of $\mathcal{A}$ from finite sets of names for states, labels and atoms and **transition axioms** of the form

```
φ ==> st -> st'              φ ==> st -> branch$sts
φ ==> (st,lab) -> st'        φ ==> (st,lab) -> branch$sts
```

Here φ is an (optional) formula (see below) and st, st', lab, sts are terms representing states (or atoms), labels and lists of states, respectively. `branch` transforms the list $[t_1, \dots, t_n]$ of terms into the term t_1`<+>`$\dots$`<+>`t_n, which is regarded as the set $\{t_1, \dots, t_n\}$ (see [138]).

The transition axioms define functions, which interpret the Σ-arrows $trans$, $transL$, $value$ and $valueL$ in $\mathcal{A}$ (see above). For accelerating the evaluation process, Expander2 works with encodings of $trans$, $transL$, $value$ and $valueL$ 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 $inits^{\mathcal{A}}$ of initial states, Expander2 iteratively applies all applicable transition axioms as rewrite rules and thus builds up Q along with the functions $trans^{\mathcal{A}}$ and $transL^{\mathcal{A}}$. Finally, $value^{\mathcal{A}}$ and $valueL^{\mathcal{A}}$ 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 $transAll(inits)^{\mathcal{A}}$ (see below). A list representation of this set is obtained by simplifying the constant `states`.

Expander2 evaluates Σ-formulas according to their above semantics by applying simplification axioms (see below). For instance, equations of the Expander2 specification `modal` define derived Σ-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:

$$\begin{aligned} eval,\ evalG &: Fo_\Sigma(V)_{\mathcal{P}(state)} \to \mathcal{P}(state), \\ evalR,\ evalRG,\ evalRM &: Fo_\Sigma(V)_{\mathcal{P}(state^2)} \to \mathcal{P}(state^2), \\ evalT,\ evalM &: Fo_\Sigma(V)_{table} \to table. \end{aligned}$$

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 Σ-algebra whose carriers consist of the *indices* of the lists of states, labels and atoms, respectively, (`states`, `labels`, `atoms`) that where created when the Kripke model is generated and whose elements are often complex terms.

Consequently, *trans* and *value* are implemented as lists of natural numbers (`tr`, `va`) and *transL* and *valueL* as lists of lists of natural numbers (`trL`, `vaL`) such that for all $i, j, k \in \mathbb{N}$,

$$\begin{array}{lcll} i \in \mathtt{tr}!!j & \Longleftrightarrow & \mathtt{states}!!i & \in trans(\mathtt{states}!!j), \\ i \in \mathtt{trL}!!j!!k & \Longleftrightarrow & \mathtt{states}!!i & \in transL(\mathtt{states}!!j)(\mathtt{labels}!!k), \\ i \in \mathtt{va}!!j & \Longleftrightarrow & \mathtt{atoms}!!i & \in value(\mathtt{atoms}!!j), \\ i \in \mathtt{vaL}!!j!!k & \Longleftrightarrow & \mathtt{atoms}!!i & \in valueL(\mathtt{atoms}!!j)(\mathtt{labels}!!k). \end{array}$$

For $\varphi : \mathcal{P}(state) \in Fo_\Sigma(V)$, $eval(\varphi)$ and $evalG(\varphi)$ compute the subset $\varphi^{\mathcal{A}}$ of Q and display it as a list resp. graph on the canvas of Expander's solver. The graph represents $trans^{\mathcal{A}}$ and $transL^{\mathcal{A}}$ with the states of $\varphi^{\mathcal{A}}$ colored green and the states of $Q \setminus \varphi^{\mathcal{A}}$ colored red.

For $r : \mathcal{P}(state^2) \in Fo_\Sigma(V)$, $evalR(r)$ and $evalRG(r)$ compute the binary relation $r^\mathcal{A}$ between states (and atoms) and display it as a list of pairs resp. the graph representing $r^\mathcal{A}$ on the canvas of Expander's solver.

$evalRM(r)$ also computes the relation $r^\mathcal{A}$, but displays the matrix representing $r^\mathcal{A}$ 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 $\vartheta : \mathcal{P}(row) \in Fo_\Sigma(V)$, $evalT(\vartheta)$ 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.

Given closed Σ-formulas $p : state \to 2$ and $at : atom$, the axiom $at \to sat(p)$ defines instances of $sat(p)$ as instances of at. Hence replacing $sat(p)$ with at in other formulas reduces the evaluation of $sat(p)$ in $\mathcal{A}$ to the direct access to $value(at)^\mathcal{A}$.

Expander2 evaluates a fixpoint formula in a finite domain D with the help of following iterative function:

$$\begin{array}{rcl} \mathit{fixpt} : \mathcal{P}(D \times D)) & \to & ((D \to D) \to (D \to D)) \\ (\leq) & \mapsto & \lambda f : D \to D.\lambda d.\mathit{if}\ f(d) \leq d\ \mathit{then}\ d\ \mathit{else}\ \mathit{fixpt}(\leq)(f)(f(d)). \end{array}$$

In particular,

$$\begin{array}{rcl} (\mu x.\varphi : \mathcal{P}(e)^{\mathcal{A}} & = & \mathit{fixpt}(\subseteq)(\Phi)(\emptyset), \\ (\nu x.\varphi : \mathcal{P}(e))^{\mathcal{A}} & = & \mathit{fixpt}(\supseteq)(\Phi)(A_e), \\ (\mu x.r : \mathcal{P}(e \times e'))^{\mathcal{A}} & = & \mathit{fixpt}(\subseteq)(\Psi)(\emptyset), \\ (\nu x.r : \mathcal{P}(e \times e'))^{\mathcal{A}} & = & \mathit{fixpt}(\supseteq)(\Psi)(A_e \times A_{e'}), \\ \sim^{\mathcal{A}} & = & \mathit{fixpt}(\supseteq)(\Psi')(Q^2) \end{array}$$

(see above).

Expander2 implements sets by lists and the rows of a table by association lists of type $(\mathit{label} \times \mathit{state})^*$.

Expander2 distinguishes between **formulas**, which agree with the characteristic functions of first-order Σ-formulas, and **terms**, which comprise all other Σ-formulas. Propositional operators $\wedge, \vee$ and quantifiers $\forall, \exists$ are denoted by `&`, `|`, `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.

Besides transition axioms there are two further kinds of axioms to be used by Expander's simplifier:

$$\varphi \texttt{ ==> t == t'}$$

$$\varphi \texttt{ ==> (}\varphi_1 \texttt{ <==> } \varphi_2\texttt{)}$$

Here t and t' are terms and $\varphi, \varphi_1, \varphi_2$ are formulas. Again, φ is optional. The axiom is applied to a formula ψ by matching t or φ_1 with a subterm resp. subformula of ψ only if the respective instance of the **guard** φ 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.

10.5 Automata for satisfiability

(see, e.g., [71, 175, 176])

Let $\mathcal{A}$ be a Σ-algebra with carrier A. Given $e \in \mathcal{T}(S)$, $a \in A_e$ **satisfies** $\varphi : \mathcal{P}(e) \in Fo_\Sigma(V)$, written as $a \models \varphi$, if $a \in \varphi^{\mathcal{A}}$.

Acceptors are used for proving that a given element $a \in A_e$ satisfies φ. φ is turned 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.,

$$Aut(\varphi) \text{ accepts } a \quad \Leftrightarrow \quad a \in \varphi^{\mathcal{A}}. \tag{1}$$

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 algebra 9.6.19, 9.6.20 and 9.6.23 and sample final algebra 9.18.6).

Then $A = \mathcal{P}(X^*)$ and $\varphi^{\mathcal{A}} = fold^{\mathcal{A}}(\varphi) \subseteq X^*$.

Since $fold^{Lang(X)} = unfold^{Bro(X)}$ (see sample biinductive definition 16.5.5), 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 Σ-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., [176], sections 1.1 and 3.3.2—, which express properties of words over X or finite $C\Sigma$-terms (see chapter 9), coincide with Σ-formulas as defined above where Σ is one of the following signatures:

$$
\begin{aligned}
MSO_{word}(X) &= (S, \emptyset, P),\\
S &= \{node\},\\
F &= \{label : node \to 1 + X,\ \ succ : node \to node\},\\
P &= \{=, <: node \times node,\ \ \in: node \times node^*\},
\end{aligned}
$$

$$
\begin{aligned}
MSO_{tree}(X) &= (S, \emptyset, P),\\
S &= \{node\},\\
F &= \{label : node \to 1 + X,\ \ children : node \to node^*,\\
P &= \{=, <: node \times node,\ \ \in: node \times node^*\}.
\end{aligned}
$$

Word acceptors

Let $w = (x_1, \ldots, x_n) \in X^*$. The $MSO_{word}(X)$-structure $\underline{w}$ is defined as follows: For all $i \in \mathbb{N}$,

$$\begin{aligned} \underline{w}_{node} &= \mathbb{N}, \\ label^{\underline{w}}(i) &= if\ 1 \leq i \leq |w|\ then\ x_i\ else\ \epsilon, \\ succ^{\underline{w}}(i) &= i+1, \\ =^{\underline{w}} &= \Delta_{\mathbb{N}}, \\ <^{\underline{w}} &= \{(i,j) \in \mathbb{N}^2 \mid i < j\}, \\ \in^{\underline{w}} &= \{(i,v) \in \mathbb{N} \times \mathbb{N}^* \mid i \in v\}. \end{aligned}$$

Let V be an S-sorted set of variables. $\mathbb{N}^V$ denotes the set of **valuations of** V, i.e., pairs $(f : V_{node} \to \mathbb{N}, g : V_{nodes} \to \mathcal{P}(\mathbb{N}))$ of functions.

$(f,g) \in \mathbb{N}^V$ induces a function $vars_{f,g} : \mathbb{N} \to \mathcal{P}(V)$ that is defined as follows: For all $i \in \mathbb{N}$,

$$vars_{f,g}(i) = \{x \in V_{node} \mid f(x) = i\} \cup \{x \in V_{nodes} \mid i \in g(x)\}.$$

Let $X_V = X \times \mathcal{P}(V)$, $\mathcal{A}$ be a nondeterministic acceptor of X_V-words, i.e., an $NAcc(X_V)$-algebra, and $s \in \mathcal{A}(state)$.

The language of words over X_V accepted by the initial automaton $(\mathcal{A}, s)$ is given by $unfold^{\mathcal{A}}(s)$ where

$$unfold^{\mathcal{A}} : \mathcal{A}(state) \to \mathcal{P}(X_V^*)$$

is the unique $NAcc(X_V)$-homomorphism from $\mathcal{A}$ to the final $NAcc(X)$-algebra $NPow(X_V)$ (see sample algebra 9.6.21). The acceptor of X_V-words becomes an acceptor of X-words by defining the **word language accepted by** $(\mathcal{A}, s)$ as follows:

$$Lan(\mathcal{A}, s) = \{w \in X^* \mid \exists\ val \in \mathbb{N}^V : h(w, val) \in unfold^{\mathcal{A}}(s)\}$$

where

$$\begin{array}{rcl} h : X^* \times \mathbb{N}^V & \to & X_V^* \\ ((x_1, \dots, x_n), (f, g)) & \mapsto & ((x_1, vars_{f,g}(1)), \dots, (x_1, vars_{f,g}(1))). \end{array}$$

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 [176], Theorem 1.18, for every $MSO_{word}(X)$-formula φ over V there is an initial automaton $(Aut(\varphi), s)$ such that for all $w \in X^*$,

$$h(w, val) \in unfold^{Aut(\varphi)}(s) \;\Leftrightarrow\; val \in \varphi^{\underline{w}}. \tag{2}$$

For the definition of $\varphi^{\underline{w}}$, the set of formulas satisfied by w, see chapter 10.

If φ is closed, then $\varphi^{\underline{w}}$ is empty or equal to $\mathbb{N}^V$. Consequently,

$$w \in Lan(Aut(\varphi), s) \;\Leftrightarrow\; \exists\, val \in \mathbb{N}^V : h(w, val) \in unfold^{Aut(\varphi)}(s) \;\stackrel{(2)}{\Leftrightarrow}\; \varphi^{\underline{w}} \neq \emptyset$$
$$\stackrel{\varphi\ is\ closed}{\Leftrightarrow}\; \varphi^{\underline{w}} = \mathbb{N}^V \;\Leftrightarrow\; \underline{w} \models \varphi.$$

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 [71], section 5, $X = \{EAT, THINK, READ\}$ and $(x_1, \ldots, x_n) \in X^*$ 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 φ 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 $\Sigma = (S, C)$ be a finitary signature and $t \in T_\Sigma$. The $MSO_{tree}(C)$-structure $\underline{t}$ is defined as follows: For all $w \in \mathbb{N}^*$,

$$
\begin{aligned}
\underline{t}_{node} &= \mathbb{N}^*, \\
label^{\underline{t}}(w) &= if\ \ w \in def(t)\ then\ t(w)\ else\ \epsilon, \\
children^{\underline{t}}(w) &= [wi \mid i \in \mathbb{N},\ wi \in def(t)], \\
=^{\underline{t}} &= \Delta_{\mathbb{N}^*}, \\
<^{\underline{t}} &= \{(v, w) \in (\mathbb{N}^*)^2 \mid \exists\, v' \in \mathbb{N}^+ : vv' = w\}, \\
\in^{\underline{t}} &= \{(v, W) \in \mathbb{N}^* \times (\mathbb{N}^*)^* \mid v \in W\}.
\end{aligned}
$$

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 $(f : V_{node} \to \mathbb{N}^*, g : V_{nodes} \to \mathcal{P}(\mathbb{N}^*))$ of functions.

$(f, g) \in (\mathbb{N}^*)^V$ induces a function $vars_{f,g} : \mathbb{N}^* \to \mathcal{P}(V)$ that is defined as follows: For all $w \in \mathbb{N}^*$,

$$vars_{f,g}(w) = \{x \in V_{node} \mid f(x) = w\} \cup \{x \in V_{nodes} \mid w \in g(x)\}.$$

Let $\Sigma_V = (S, \{(c, V') : e \to s \mid c : e \to s,\ V' \subseteq V\})$ and $\mathcal{A}$ be a nondeterministic top-down acceptor of ground Σ-terms, i.e., an $NTAcc(\Sigma_V)$-algebra, and $s \in \mathcal{A}(state)$.

The language of Σ_V-terms accepted by the initial automaton $(\mathcal{A}, s)$ is given by $unfold^{\mathcal{A}}(s)$ where

$$unfold^{\mathcal{A}} : \mathcal{A}(state) \to \mathcal{P}(T_{\Sigma_V})$$

is the unique $NTAcc(\Sigma_V)$-homomorphism from $\mathcal{A}$ to the final $NTAcc(\Sigma_V)$-algebra $NTPow(\Sigma_V)$ (see sample algebra 9.6.30). The acceptor of Σ_V-terms becomes an acceptor of Σ-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

$$\begin{array}{rcl} h : T_\Sigma \times \mathbb{N}^V & \to & T_{\Sigma_V} \\ (t, (f, g)) & \mapsto & \lambda w.(t(w), vars_{f,g}(w)). \end{array}$$

The actual goal is to use $(\mathcal{A}, s)$ as an automaton that accepts a Σ-term t iff the $MSO_{tree}(C)$-structure $\underline{t}$ defined above satisfies some given $MSO_{tree}(C)$-formula. Indeed, by [176], Theorem 3.58, for every $MSO_{tree}(C)$-formula φ over V there is an initial automaton $(Aut(\varphi), s)$ such that for all $t \in T_\Sigma$,

$$h(t, val) \in unfold^{\mathcal{A}}(s) \iff val \in \varphi^{\underline{t}}. \tag{3}$$

For the definition of $\varphi^{\underline{t}}$, the set of formulas satisfied by t, see chapter 10.

If φ is closed, then $\varphi^{\underline{t}}$ is empty or equal to $(\mathbb{N}^*)^V$. Consequently,

$$t \in Lan(Aut(\varphi), s) \;\Leftrightarrow\; \exists\, val \in (\mathbb{N}^*)^V : h(t, val) \in unfold^{Aut(\varphi)}(s) \;\overset{(3)}{\Leftrightarrow}\; \varphi^{\underline{t}} \neq \emptyset$$

$$\overset{\varphi\ is\ closed}{\Leftrightarrow}\; \varphi^{\underline{t}} = (\mathbb{N}^*)^V \;\Leftrightarrow\; \underline{t} \models \varphi.$$

10.6 Institutions

An **institution** (see [54]) consists of

- a category $Sign$ of signatures,
- a functor

$$\begin{array}{rcl} Sen : Sign & \to & Set \\ \Sigma & \mapsto & \text{set of } \Sigma\text{-sentences} \\ \sigma : \Sigma \to \Sigma' & \mapsto & Sen(\sigma) : Sen(\Sigma) \to Sen(\Sigma'), \end{array}$$

- a functor

$$\begin{array}{rcl} Mod : Sign & \to & Cat^{op} \\ \Sigma & \mapsto & \text{category of } \Sigma\text{-models} \\ \sigma : \Sigma \to \Sigma' & \mapsto & Mod(\sigma) : Mod(\Sigma') \to Mod(\Sigma), \end{array}$$

- for each $\Sigma \in Sign$, a satisfaction relation

$$\models_\Sigma \subseteq Mod(\Sigma) \times Sen(\Sigma)$$

  such that for all $Sign$-morphisms $\sigma : \Sigma \to \Sigma'$, $\mathcal{A} \in Mod(\Sigma')$ and $\varphi \in Sen(\Sigma)$.

$$Mod(\sigma)(\mathcal{A}) \models_\Sigma \varphi \quad \Leftrightarrow \quad \mathcal{A} \models_{\Sigma'} Sen(\sigma)(\varphi). \tag{1}$$

Suppose that

- $Sign$ is the category of signatures and signature morphisms as defined in chapter 9,
- for all signatures Σ, $Sen(\Sigma)$ is the set of Σ-formulas over a fixed set of variables,
- for all signature morphisms $\sigma : \Sigma \to \Sigma'$ and Σ-formulas φ, $Sen(\sigma)$ maps φ to $\sigma(\varphi)$ where $\sigma(\varphi)$ is obtained from φ by replacing all arrows of Σ by their σ-images,
- for all signatures Σ, $Mod(\Sigma) = Alg_\Sigma$,
- for all signature morphisms $\sigma : \Sigma \to \Sigma'$ and Σ′-algebras $\mathcal{A}$, $Mod(\sigma)$ maps $\mathcal{A}$ to $\mathcal{A}|_\sigma$ (see chapter 9),
- $\models$ is the satisfaction relation defined in section 10.3.

$(Sign, Sen, Mod, \models)$ is an institution.

Proof. (1) amounts to:

$$\mathcal{A}|_\sigma \models_\Sigma \varphi \quad \Leftrightarrow \quad \mathcal{A} \models_{\Sigma'} \sigma(\varphi). \tag{2}$$

The proof of (2) is straightforward (induction on the size of φ). ❏

11.1 Syntax and semantics

Let (S, F) be a signature, $\mathcal{C}$ be an (S, F)-algebra with carrier A, P be a set of **predicates**, i.e., arrows $p : e' \to \mathcal{P}(e)$ for some $e, e' \in \mathcal{T}(S)$, and $\Sigma = (S, F \cup P)$.

Struct$_{\Sigma,\mathcal{C}}$ denotes the category of Σ-algebras $\mathcal{A}$ and Σ-homomorphisms with $\mathcal{A}|_{(S,F)} = \mathcal{C}$.

Struct$_{\Sigma,\mathcal{C}}$ is a complete Boolean algebra with the following partial order $\leq$, least element $\bot$, greatest element $\top$, complements $\overline{\mathcal{A}}$, suprema $\bigsqcup \mathcal{K}$ and infima $\bigsqcap \mathcal{K}$:

For all $\mathcal{A}, \mathcal{B} \in Struct_{\Sigma,\mathcal{C}}$, $\mathcal{K} \subseteq Struct_{\Sigma,\mathcal{C}}$, $p : e' \to \mathcal{P}(e) \in P$ and $b \in A_{e'}$,

$$\begin{array}{l} \mathcal{A} \leq \mathcal{B} \;\;\Leftrightarrow\;\; \forall\, p : e' \to \mathcal{P}(e) \in P,\; b \in A_{e'} : p^{\mathcal{A}}(b) \subseteq p^{\mathcal{B}}(b), \\ p^{\bot}(b) = \emptyset, \quad p^{\top}(b) = A_e, \quad p^{\overline{\mathcal{A}}}(b) = A_e \setminus p^{\mathcal{A}}(b), \\ p^{\bigsqcup \mathcal{K}}(b) = \bigcup_{\mathcal{A} \in \mathcal{K}} p^{\mathcal{A}}(b), \quad p^{\bigsqcup \mathcal{K}}(b) = \bigcap_{\mathcal{A} \in \mathcal{K}} p^{\mathcal{A}}(b). \end{array}$$

Let φ be a Σ-formula that is negation-free up to F (see section 10.2). Then the semantics of φ is monotone w.r.t. the above partial order on *Struct*$_{\Sigma,\mathcal{C}}$, i.e., for all $\mathcal{A}, \mathcal{B} \in Struct_{\Sigma,\mathcal{C}}$,

$$\mathcal{A} \leq \mathcal{B} \quad \text{implies} \quad \varphi^{\mathcal{A}} \subseteq \varphi^{\mathcal{B}}. \tag{1}$$

(1) can be shown by induction on the size of φ.

A Σ-formula φ is **flat for** P if φ is negation-free up to F and every atom at does not contain symbols of P or $at = p(u)(t)$ for some $p \in P$ and $t, u \in \Lambda_{(S,F)}(V)$.

A Σ-formula $\varphi \Leftarrow \psi$ or $\varphi \Rightarrow \psi$ is called a **Σ-sequent for** P if φ and ψ are flat for P.

Given $p : e' \to \mathcal{P}(e) \in P$ and $t : e,\ u : e' \in \Lambda_\Sigma(V)$, a Σ-sequent $p(u)(t) \Leftarrow \varphi$ is called a **Horn clause for** p, while $p(u)(t) \Rightarrow \varphi$ is called a **co-Horn clause for** p.

Given $f : e \to e' \in F$ with $e' \neq 2$ and $t : e,\ t' : e' \in \Lambda_\Sigma(V)$, a Σ-sequent $f(t) = t' \Leftarrow \varphi$ is called a **Horn clause for** f.

Given a "transition relation" $\to: e \times e \to 2 \in F$ and $t : e,\ t' : e \in \Lambda_\Sigma(V)$, a Σ-sequent $t \to t' \Leftarrow \varphi$ is called a **Horn clause for** $\to$.

The premise of a Horn or co-Horn clause is sometimes splitted into a **guard** (to be proved before the rule is applied) and the rest an instance of which is part of the rule reduct (see section 11.5).

Let Σ and $\mathcal{C}$ be as above, AX be a set of Σ-sequents and $SP = (\Sigma, AX, \mathcal{C})$.

SP is a **Horn specification of** P and the elements of P are called **least predicates** if AX consists of Horn clauses for P.

SP is a **co-Horn specification of** P and the predicates of P are called **greatest predicates** if AX consists of co-Horn clauses for P.

$Struct_{SP}$ denotes the full subcategory of $Struct_{\Sigma,\mathcal{C}}$ whose objects satisfy AX.

The **step function** $\Phi_{SP} : Struct_{\Sigma,\mathcal{C}} \to Struct_{\Sigma,\mathcal{C}}$ is defined as follows:

For all $\mathcal{A} \in Struct_{\Sigma,\mathcal{C}}$, $p : e' \to \mathcal{P}(e) \in P$ and $b \in A_{e'}$,

$$p^{\Phi_{SP}(\mathcal{A})}(b) = \{a \in A_e \mid \left\{\begin{array}{c} \exists\ p(u)(t) \Leftarrow \varphi \in AX,\ g \in \varphi^{\mathcal{A}} : \langle t, u\rangle^{\mathcal{C}}(g) = (a, b) \\ \text{if } SP \text{ is a Horn specification,} \\ \forall\ p(u)(t) \Rightarrow \varphi \in AX,\ g \in A^V \setminus \varphi^{\mathcal{A}} : \langle t, u\rangle^{\mathcal{C}}(g) \neq (a, b) \\ \text{if } SP \text{ is a co-Horn specification.} \end{array}\right\}\} \tag{2}$$

By (1), Φ_{SP} is monotone and thus by Theorem 3.9 (1) and (5), Φ_{SP} has the least fixpoint

$$lfp(\Phi_{SP}) \;=\; \bigsqcap\{\mathcal{A} \in Struct_{\Sigma,\mathcal{C}} \mid \Phi_{SP}(\mathcal{A}) \leq \mathcal{A}\} \tag{3}$$

if SP is a Horn specification, while Φ_{SP} has the greatest fixpoint

$$gfp(\Phi_{SP}) \;=\; \bigsqcup\{\mathcal{A} \in Struct_{\Sigma,\mathcal{C}} \mid \mathcal{A} \leq \Phi_{SP}(\mathcal{A})\} \tag{4}$$

if SP is a co-Horn specification.

Let SP be a Horn specification of P, $p : e' \to \mathcal{P}(e) \in P$, AX_p be the set of Horn clauses of AX for p, $x \in V_e \setminus var(AX_p)$ and $z \in V_{e'} \setminus var(AX_p)$. The Σ-formula

$$[AX_p] \quad =_{def} \quad p(z)(x) \Leftrightarrow \bigvee_{(p(u)(t) \Leftarrow \varphi) \in AX} \exists\, free(\varphi) : ((t,u) = (x,z) \wedge \varphi)$$

is called the **Horn completion of** AX_p.

As $[AX_p]$ combines all Horn clauses of AX for p to a single Σ'-formula, the sum extension

$$[c]_{c:e \to s \in C} : \coprod_{c:e \to s \in C} \to s$$

induced by a constructive signature $\Sigma = (S, C)$ combines all arrows of C with target s to a single one (see section 15.1). This analogy strongly resembles the Curry-Howard correspondence between formulas and types.

Let SP be a co-Horn specification of P, $p : e' \to \mathcal{P}(e) \in P$, AX_p be the set of co-Horn clauses of AX for p, $x \in V_e \setminus var(AX_p)$ and $z \in V_{e'} \setminus var(AX_p)$. The Σ-formula

$$\langle AX_p \rangle \quad =_{def} \quad p(z)(x) \Leftrightarrow \bigwedge_{(p(u)(t) \Rightarrow \varphi) \in AX} \forall\, free(\varphi) : ((t,u) \neq (x,z) \vee \varphi)$$

is called the **co-Horn completion of** AX_p.

As $\langle AX_p\rangle$ combines all co-Horn clauses of AX for p to a single Σ'-formula, the product extension

$$\langle d\rangle_{d:s\to e\in D} : s \to \prod_{d:s\to e\in D}$$

induced by a destructive signature $\Sigma = (S, D)$ combines all arrows of D with source s to a single one (see section 15.2). Again, the analogy resembles the Curry-Howard correspondence between formulas and types.

Lemma 11.1 Suppose that SP is a Horn specification of P and $\Phi = \Phi_{SP}$.

$$Struct_{SP} = \{\mathcal{A} \in Struct_{\Sigma,\mathcal{C}} \mid \Phi(\mathcal{A}) \leq \mathcal{A}\}. \tag{5}$$

Moreover, for all $\mathcal{A} \in Struct_{SP}$,

$$lfp(\Phi) \leq \mathcal{A}, \tag{6}$$

$$\Phi(\mathcal{A}) = \mathcal{A} \quad \text{iff} \quad \forall\, p \in P : \mathcal{A} \models [AX_p]. \tag{7}$$

Proof. (5) Let $p : e' \to \mathcal{P}(e) \in P$, $\mathcal{A} \in Struct_{SP}$ with carrier A, $b \in A_{e'}$ and $a \in p^{\Phi(\mathcal{A})}(b)$. Then by (2), $(a, b) = \langle t, u\rangle^{\mathcal{C}}(g)$ for some $p(u)(t) \Leftarrow \varphi \in AX$ and $g \in \varphi^{\mathcal{A}}$. Since $\mathcal{A}$ satisfies $p(u)(t) \Leftarrow \varphi$, $g \in p(u)(t)^{\mathcal{A}}$ and thus $a = t^{\mathcal{C}}(g) \in p(u)^{\mathcal{A}}(g) = p^{\mathcal{A}}(u^{\mathcal{C}}(g)) = p^{\mathcal{A}}(b)$. Hence $p^{\Phi(\mathcal{A})}(b) \subseteq p^{\mathcal{A}}(b)$ and thus $\Phi(\mathcal{A}) \leq \mathcal{A}$.

Conversely, let $\Phi(\mathcal{A}) \le \mathcal{A}$, $p(u)(t) \Leftarrow \varphi \in AX$ and $g \in \varphi^{\mathcal{A}}$. Then by (2), $t^{\mathcal{C}}(g) \in p^{\Phi(\mathcal{A})}(u^{\mathcal{C}}(g))$. Since $\Phi(\mathcal{A}) \le \mathcal{A}$, $t^{\mathcal{C}}(g) \in p^{\mathcal{A}}(u^{\mathcal{C}}(g))$ and thus $g \in p(u)(t)^{\mathcal{A}}$. Therefore, $\mathcal{A}$ satisfies $p(u)(t) \Leftarrow \varphi$.

(6) For all $\mathcal{A} \in \mathit{Struct}_{SP}$, $p : e' \to \mathcal{P}(e) \in P$, $a \in A_e$ and $b \in A_{e'}$,

$$a \in p^{\mathit{lfp}(\Phi)}(b) \stackrel{(3)}{\Rightarrow}$$

$$\forall \mathcal{B} \in \mathit{Struct}_{\Sigma,\mathcal{C}} : (\Phi(\mathcal{B}) \le \mathcal{B} \Rightarrow a \in p^{\mathcal{B}}(b)) \stackrel{(5)}{\Rightarrow} \forall \mathcal{B} \in \mathit{Struct}_{SP} : a \in p^{\mathcal{A}}(b),$$

i.e., $\mathit{lfp}(\Phi) \le \mathcal{A}$.

(7) Let $\mathcal{A} \in \mathit{Struct}_{SP}$. Since $\Phi(\mathcal{A}) = \mathcal{A}$ iff for all $p \in P$, $p^{\Phi(\mathcal{A})} = p^{\mathcal{A}}$, it remains to show that for all $p : e' \to \mathcal{P}(e) \in P$,

$$p^{\Phi(\mathcal{A})} = p^{\mathcal{A}} \;\Leftrightarrow\; \mathcal{A} \models [AX_p]. \tag{8}$$

Let $x \in V_e \setminus var(AX_p)$ and $z \in V_{e'} \setminus var(AX_p)$. Then

$$\begin{aligned} & p^{\Phi(\mathcal{A})} = p^{\mathcal{A}} \\ \stackrel{(2)}{\Leftrightarrow}\; & \forall\, a \in A_e, b \in A_{e'} : \\ & \qquad a \in p^{\mathcal{A}}(b) \Leftrightarrow \exists\, p(u)(t) \Leftarrow \varphi \in AX, g \in \varphi^{\mathcal{A}} : \langle t,u\rangle^{\mathcal{C}}(g) = (a,b) \\ \Leftrightarrow\; & p(z)(x)^{\mathcal{A}} = \textstyle\bigcup_{p(u)(t)\Leftarrow\varphi\in AX}\{g \in \varphi^{\mathcal{A}} \mid \langle t,u\rangle^{\mathcal{C}}(g) = (g(x),g(z))\} \end{aligned}$$

$$\begin{aligned}
&\Leftrightarrow\ p(z)(x)^{\mathcal{A}} = \bigcup\nolimits_{p(u)(t)\Leftarrow\varphi\in AX}(\exists\, \mathit{free}(\varphi) : ((t,u) = (x,z) \wedge \varphi))^{\mathcal{A}} \\
&\Leftrightarrow\ p(z)(x)^{\mathcal{A}} = (\bigvee\nolimits_{p(u)(t)\Leftarrow\varphi\in AX} \exists\, \mathit{free}(\varphi) : ((t,u) = (x,z) \wedge \varphi))^{\mathcal{A}} \\
&\Leftrightarrow\ (p(z)(x) \Leftrightarrow (\bigvee\nolimits_{p(u)(t)\Leftarrow\varphi\in AX} \exists\, \mathit{free}(\varphi) : ((t,u) = (x,z) \wedge \varphi)))^{\mathcal{A}} = A^V \\
&\Leftrightarrow\ [AX_p]^{\mathcal{A}} = A^V \\
&\Leftrightarrow\ \mathcal{A} \models [AX_p],
\end{aligned}$$

i.e., (8) holds true. ❑

Lemma 11.1 (5) implies $\mathit{lfp}(\Phi_{SP}) \in \mathit{Struct}_{SP}$. Hence Lemma 11.1 (6) justifies it to call $\mathit{lfp}(\Phi_{SP})$ the **least solution of AX in $\mathit{Struct}_{\Sigma,\mathcal{C}}$**.

Lemma 11.2 Suppose that SP is a co-Horn specification of P and $\Phi = \Phi_{SP}$.

$$\mathit{Struct}_{SP} = \{\mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}} \mid \mathcal{A} \leq \Phi(\mathcal{A})\}. \tag{9}$$

Moreover, for all $\mathcal{A} \in \mathit{Struct}_{SP}$,

$$\mathcal{A} \leq \mathit{gfp}(\Phi), \tag{10}$$

$$\Phi(\mathcal{A}) = \mathcal{A} \quad \text{iff} \quad \forall\, p \in P : \mathcal{A} \models \langle AX_p \rangle. \tag{11}$$

Proof. (9) Let $p : e' \to \mathcal{P}(e) \in P$, $\mathcal{A} \in \mathit{Struct}_{SP}$ with carrier A, $b \in A_{e'}$ and $a \in A_e \setminus p^{\Phi(\mathcal{A})}(b)$. Then by (2), $(a,b) = \langle t,u\rangle^{\mathcal{C}}(g)$ for some $p(u)(t) \Rightarrow \varphi \in AX$ and $g \in A^V \setminus \varphi^{\mathcal{A}}$.

Since $\mathcal{A}$ satisfies $p(u)(t) \Rightarrow \varphi$, $g \in A^V \setminus p(u)(t)^{\mathcal{A}}$ and thus

$$a = t^{\mathcal{C}}(g) \in A_e \setminus p(u)^{\mathcal{A}}(g) = A_e \setminus p^{\mathcal{A}}(u^{\mathcal{C}}(g)) = A_e \setminus p^{\mathcal{A}}(b).$$

Hence $p^{\mathcal{A}}(b) \subseteq p^{\Phi(\mathcal{A})}(b)$ and thus $\mathcal{A} \leq \Phi(\mathcal{A})$.

Conversely, let $\mathcal{A} \leq \Phi(\mathcal{A})$, $p(u)(t) \Rightarrow \varphi \in AX$ and $g \in A^V \setminus \varphi^{\mathcal{A}}$. Then by (2), $t^{\mathcal{C}}(g) \subseteq A_e \setminus p^{\Phi(\mathcal{A})}(u^{\mathcal{C}}(g))$. Since $\mathcal{A} \leq \Phi(\mathcal{A})$, $t^{\mathcal{C}}(g) \in A_e \setminus p^{\mathcal{A}}(u^{\mathcal{C}}(g))$ and thus $g \in A^V \setminus p(u)(t)^{\mathcal{A}}$. Therefore, $\mathcal{A}$ satisfies $p(u)(t) \Rightarrow \varphi$.

(10) For all $\mathcal{A} \in \mathit{Struct}_{SP}$, $p : e' \to \mathcal{P}(e) \in P$, $a \in A_e$ and $b \in A_{e'}$,

$$\begin{aligned} a \in p^{\mathcal{A}}(b) \;&\Rightarrow\; \exists \mathcal{B} \in \mathit{Struct}_{SP} : a \in p^{\mathcal{B}}(b) \\ &\overset{(9)}{\Rightarrow}\; \exists \mathcal{B} \in \mathit{Struct}_{\Sigma,\mathcal{C}} : (\mathcal{B} \leq \Phi(\mathcal{B}) \wedge a \in p^{\mathcal{B}}(b)) \;\overset{(4)}{\Rightarrow}\; a \in p^{gfp(\Phi)}(b), \end{aligned}$$

i.e., $\mathcal{A} \leq gfp(\Phi)$.

(11) Let $\mathcal{A} \in \mathit{Struct}_{SP}$. Since $\Phi(\mathcal{A}) = \mathcal{A}$ iff for all $p : e' \to \mathcal{P}(e) \in P$, $p^{\Phi(\mathcal{A})} = p^{\mathcal{A}}$, it remains to show that for all $p \in P$,

$$p^{\Phi(\mathcal{A})} = p^{\mathcal{A}} \;\Leftrightarrow\; \mathcal{A} \models \langle AX_p \rangle. \tag{12}$$

Let $x \in V_e \setminus var(AX_p)$ and $z \in V_{e'} \setminus var(AX_p)$. Then

$$
\begin{array}{ll}
 & p^{\Phi(\mathcal{A})} = p^{\mathcal{A}} \\
\stackrel{(2)}{\Leftrightarrow} & \forall\ a \in A_e, b \in A_{e'} : \\
 & \quad a \in p^{\mathcal{A}}(b) \Leftrightarrow \forall\ p(u)(t) \Rightarrow \varphi \in AX, g \in A^V \setminus \varphi^{\mathcal{A}} : \langle t,u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
\Leftrightarrow & p(z)(x)^{\mathcal{A}} = \bigcap_{p(u)(t)\Rightarrow\varphi\in AX}\{g \in A^V \mid \langle t,u\rangle^{\mathcal{C}}(g) \neq (g(x),g(z)) \vee g \in \varphi^{\mathcal{A}}\} \\
\Leftrightarrow & p(z)(x)^{\mathcal{A}} = \bigcap_{p(u)(t)\Rightarrow\varphi\in AX}(\forall\ free(\varphi) : ((t,u) \neq (x,z) \vee \varphi))^{\mathcal{A}} \\
\Leftrightarrow & p(z)(x)^{\mathcal{A}} = (\bigwedge_{p(u)(t)\Rightarrow\varphi\in AX} \forall\ free(\varphi) : ((t,u) \neq (x,z) \vee \varphi))^{\mathcal{A}} \\
\Leftrightarrow & (p(z)(x) \Leftrightarrow (\bigwedge_{p(u)(t)\Rightarrow\varphi\in AX} \forall\ free(\varphi) : ((t,u) \neq (x,z) \vee \varphi)))^{\mathcal{A}} = A^V \\
\Leftrightarrow & \langle AX_p\rangle^{\mathcal{A}} = A^V \\
\Leftrightarrow & \mathcal{A} \models \langle AX_p\rangle,
\end{array}
$$

i.e., (12) holds true. ❑

Lemma 11.2 (9) implies $gfp(\Phi_{SP}) \in Struct_{SP}$. Hence Lemma 11.2 (10) justifies it to call $gfp(\Phi_{SP})$ the **greatest solution of AX in $Struct_{\Sigma,\mathcal{C}}$**.

Theorem 11.3 (How to specify complement predicates)

Let $SP = (\Sigma, AX, \mathcal{C})$, $\Phi = \Phi_{SP}$,

$$AX' = \begin{cases} \{p(u)(t) \Rightarrow \overline{\varphi} \mid p(u)(t) \Leftarrow \varphi \in AX\} & \text{if } SP \text{ is a Horn specification,} \\ \{p(u)(t) \Leftarrow \overline{\varphi} \mid p(u)(t) \Rightarrow \varphi \in AX\} & \text{if } SP \text{ is a co-Horn specification,} \end{cases}$$

$SP' = (\Sigma', AX', \mathcal{C})$ and $\Phi' = \Phi_{SP'}$ where $\overline{\varphi}$ is defined inductively as follows:

$\overline{True} = False$, $\overline{False} = True$, for all atoms $at \in \Lambda_{(S,F)}(V)$, $\overline{at} = \neg at$, for all $p : e' \to \mathcal{P}(e) \in P$ and $t : e$, $u : e' \in \Lambda_{(S,F)}(V)$, $\overline{p(u)(t)} = p(u)(t)$, and for all $\varphi, \psi \in Fo_{\Sigma'}(V)$ and $x \in V$, $\overline{\varphi \wedge \psi} = \overline{\varphi} \vee \overline{\psi}$, $\overline{\varphi \vee \psi} = \overline{\varphi} \wedge \overline{\psi}$, $\overline{\forall x \varphi} = \exists x \overline{\varphi}$ and $\overline{\exists x \varphi} = \forall x \overline{\varphi}$.

(i) Let SP be a Horn specification of P. $gfp(\Phi') = \overline{lfp(\Phi)}$, i.e., for all $p : e' \to \mathcal{P}(e) \in P$ and $b \in A_{e'}$,

$$p^{gfp(\Phi')}(b) = A_e \setminus p^{lfp(\Phi)}(b).$$

(ii) Let SP be a co-Horn specification of P. $lfp(\Phi') = \overline{gfp(\Phi)}$, i.e., for all $p : e' \to \mathcal{P}(e) \in P$ and $b \in A_{e'}$,

$$p^{lfp(\Phi')}(b) = A_e \setminus p^{gfp(\Phi)}(b).$$

Proof. (i) Following the proof of Theorem 3.10, suppose that for all $\mathcal{A} \in Struct_{\Sigma,\mathcal{C}}$,

$$\Phi'(\mathcal{A}) = \overline{\Phi(\overline{\mathcal{A}})}. \tag{13}$$

Then

$$\begin{aligned}
gfp(\Phi') &\stackrel{(4)}{=} \bigsqcup\{\mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}} \mid \mathcal{A} \leq \Phi'(\mathcal{A})\} \stackrel{(13)}{=} \bigsqcup\{\mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}} \mid \mathcal{A} \leq \overline{\Phi(\overline{\mathcal{A}})}\} \\
&= \bigsqcup\{\mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}} \mid \Phi(\overline{\mathcal{A}}) \leq \overline{\mathcal{A}}\} = \bigsqcup\{\overline{\mathcal{A}} \mid \mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}},\ \Phi(\overline{\overline{\mathcal{A}}}) \leq \overline{\overline{\mathcal{A}}}\} \\
&= \bigsqcup\{\overline{\mathcal{A}} \mid \mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}},\ \Phi(\mathcal{A}) \leq \mathcal{A}\} = \overline{\bigsqcap\{\mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}} \mid \Phi(\mathcal{A}) \leq \mathcal{A}\}} = \overline{lfp(\Phi)}.
\end{aligned}$$

It remains to show (13). Suppose that for all $\mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}}$ and Σ-formulas φ that are flat for P,

$$\overline{\varphi}^{\mathcal{A}} \;=\; A^V \setminus \varphi^{\overline{\mathcal{A}}}. \tag{14}$$

Then for all $p : e' \to \mathcal{P}(e) \in P$, $\mathcal{A} \in \mathit{Struct}_{\Sigma,\mathcal{C}}$, $a \in A_e$ and $b \in A_{e'}$,

$$\begin{aligned}
& a \in p^{\overline{\Phi(\overline{\mathcal{A}})}}(b) \;\Leftrightarrow\; a \notin p^{\Phi(\overline{\mathcal{A}})}(b) \\
&\stackrel{(2)}{\Leftrightarrow}\; \neg(\exists\, p(u)(t) \Leftarrow \varphi \in AX,\ g \in \varphi^{\overline{\mathcal{A}}} : \langle t,u\rangle^{\mathcal{C}}(g) = (a,b)) \\
&\Leftrightarrow\; \forall\, p(u)(t) \Leftarrow \varphi \in AX,\ g \in \varphi^{\overline{\mathcal{A}}} : \langle t,u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
&\Leftrightarrow\; \forall\, p(u)(t) \Leftarrow \varphi \in AX,\ g \in A^V \setminus A^V \setminus \varphi^{\overline{\mathcal{A}}} : \langle t,u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
&\stackrel{(14)}{\Leftrightarrow}\; \forall\, p(u)(t) \Rightarrow \overline{\varphi} \in AX',\ g \in A^V \setminus \overline{\varphi}^{\mathcal{A}} : \langle t,u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
&\stackrel{(2)}{\Leftrightarrow}\; a \in p^{\Phi'(\mathcal{A})}.
\end{aligned}$$

It remains to show (14). We show (14) by induction on the size of φ:

$$\overline{True}^{\mathcal{A}} = False^{\mathcal{A}} = \emptyset = A^V \setminus A^V = A^V \setminus True^{\overline{\mathcal{A}}},$$
$$\overline{False}^{\mathcal{A}} = True^{\mathcal{A}} = A^V = A^V \setminus \emptyset = A^V \setminus False^{\overline{\mathcal{A}}}.$$

For all atoms $at \in \Lambda_{(S,F)}(V)$ and $g \in A^V$,

$$g \in \overline{at}^{\mathcal{A}} \;\Leftrightarrow\; g \in (\neg at)^{\mathcal{A}} \;\Leftrightarrow\; g \in (\neg at)^{\mathcal{C}} \;\Leftrightarrow\; g \notin at^{\mathcal{C}} \Leftrightarrow\; g \notin at^{\overline{\mathcal{A}}} \;\Leftrightarrow\; g \in A^V \setminus at^{\overline{\mathcal{A}}}.$$

For all $p : e' \to \mathcal{P}(e) \in P$, $t : e,\ u : e' \in \Lambda_{(S,F)}(V)$ and $g \in A^V$,

$$g \in \overline{p(u)(t)}^{\mathcal{A}} \;\Leftrightarrow\; g \in p(u)(t)^{\mathcal{A}} \;\Leftrightarrow\; t^{\mathcal{C}}(g) \in p(u)^{\mathcal{A}} \;\Leftrightarrow\; t^{\mathcal{C}}(g) \notin p(u)^{\overline{\mathcal{A}}}$$
$$\Leftrightarrow\; g \notin p(u)(t)^{\overline{\mathcal{A}}} \;\Leftrightarrow\; g \in A^V \setminus p(u)(t)^{\overline{\mathcal{A}}}.$$

For all $\varphi, \psi \in Fo_{\Sigma'}(V)$ and $x \in V$,

$$\overline{\varphi \wedge \psi}^{\mathcal{A}} = (\overline{\varphi} \vee \overline{\psi})^{\mathcal{A}} = \overline{\varphi}^{\mathcal{A}} \cup \overline{\psi}^{\mathcal{A}} \stackrel{ind.\ hyp.}{=} A^V \setminus \varphi^{\overline{\mathcal{A}}} \cup A^V \setminus \psi^{\overline{\mathcal{A}}} = A^V \setminus (\varphi^{\overline{\mathcal{A}}} \cap \psi^{\overline{\mathcal{A}}})$$
$$= A^V \setminus (\varphi \wedge \psi)^{\overline{\mathcal{A}}},$$
$$\overline{\varphi \vee \psi}^{\mathcal{A}} = (\overline{\varphi} \wedge \overline{\psi})^{\mathcal{A}} = \overline{\varphi}^{\mathcal{A}} \cap \overline{\psi}^{\mathcal{A}} \stackrel{ind.\ hyp.}{=} A^V \setminus \varphi^{\overline{\mathcal{A}}} \cap A^V \setminus \psi^{\overline{\mathcal{A}}} = A^V \setminus (\varphi^{\overline{\mathcal{A}}} \cup \psi^{\overline{\mathcal{A}}})$$
$$= A^V \setminus (\varphi \vee \psi)^{\overline{\mathcal{A}}},$$
$$\overline{\forall x \varphi}^{\mathcal{A}} = (\exists x \overline{\varphi})^{\mathcal{A}} = \bigcup_{a \in A_e} \{g \in A^V \mid g[a/x] \in \overline{\varphi}^{\mathcal{A}}\}$$
$$\stackrel{ind.\ hyp.}{=} \bigcup_{a \in A_e} \{g \in A^V \mid g[a/x] \in A^V \setminus \varphi^{\overline{\mathcal{A}}}\} = A^V \setminus \bigcap_{a \in A_e} \{g \in A^V \mid g[a/x] \in \varphi^{\overline{\mathcal{A}}}\}$$
$$= A^V \setminus (\forall x \varphi)^{\overline{\mathcal{A}}},$$

$$\overline{\exists x\varphi}^{\mathcal{A}} = (\forall x\overline{\varphi})^{\mathcal{A}} = \bigcap_{a\in A_e}\{g \in A^V \mid g[a/x] \in \overline{\varphi}^{\mathcal{A}}\}$$
$$\stackrel{ind.\ hyp.}{=} \bigcap_{a\in A_e}\{g \in A^V \mid g[a/x] \in A^V \setminus \varphi^{\overline{\mathcal{A}}}\} = A^V \setminus \bigcup_{a\in A_e}\{g \in A^V \mid g[a/x] \in \varphi^{\overline{\mathcal{A}}}\}$$
$$= A^V \setminus (\exists x\varphi)^{\overline{\mathcal{A}}}$$

(ii) can be shown analogously. ❏

In the following examples, atoms of the form $p()(t)$ are abbreviated to $p(t)$.

Example 1 (even and odd) Let $S = \{nat\}$,

$$\begin{array}{rcl} F &=& \{0 : 1 \to nat, succ : nat \to nat\}, \\ \mathcal{C} &=& \mathbb{N}, \qquad \text{(see sample algebra 9.6.1)} \\ P &=& \{even, odd : 1 \to \mathcal{P}(nat)\}, \quad \Sigma = (S, F \cup P), \quad V = \{n : nat\} \end{array}$$

and AX consist of the following Horn clauses:

$$\begin{array}{rcl} even(0) &\Leftarrow& True \\ even(succ(n)) &\Leftarrow& odd(n) \\ odd(succ(n)) &\Leftarrow& even(n) \end{array}$$

The least solution of AX in $Struct_{\Sigma,\mathcal{C}}$ interprets *even* and *odd* as the sets of even and odd natural numbers, respectively.

Example 2 (partition and flatten lists) Let X be a set, $S = \{state, state'\}$,

$$\begin{aligned} F &= \{\alpha : 1 \to state,\ \alpha' : 1 \to state',\ cons : X \times state \to state, \\ &\quad\ cons' : state \times state' \to state',\ \ ++ : state \times state \to state\}, \\ \mathcal{C} &= X^*, \qquad\qquad \text{(see sample algebra 9.6.3)} \\ P &= \{part : 1 \to \mathcal{P}(state \times state'),\ flatten : 1 \to \mathcal{P}(state' \times state)\}, \end{aligned}$$

$\Sigma = (S, F \cup P)$, $V = \{x, y : X,\ s, s' : state,\ p : state'\}$ and AX consist of the following Horn clauses:

$$\begin{aligned} part(cons(x,\alpha), cons'(cons(\alpha),\alpha')) &\Leftarrow True \\ part(cons(x,cons(y,s)), cons'(cons(x,\alpha),p)) &\Leftarrow part(cons(y,s),p) \\ part(cons(x,cons(y,s)), cons'(cons(x,s'),p)) &\Leftarrow part(cons(y,s),cons'(s',p)) \\ flatten(\alpha',\alpha) &\Leftarrow True \\ flatten(cons'(s,p), s +\!\!+ s') &\Leftarrow flatten(p,s') \end{aligned}$$

The least solution of AX in $Struct_{\Sigma,\mathcal{C}}$ interprets *part* and *flatten* as the I/O relations of partitioning and flattening lists, respectively.

Example 3 (sorted and unsorted) Let X be a set, $S = \{state\}$,

$$\begin{aligned} F &= \{\alpha : 1 \to state,\ cons : X \times state \to state,\ \ \leq : 1 \to \mathcal{P}(X \times X)\}, \\ \mathcal{C} &= X^*, \qquad \text{(see sample algebra 9.6.3)} \\ P &= \{sorted, unsorted : 1 \to \mathcal{P}(state)\}, \end{aligned}$$

$\Sigma = (S, F \cup P)$, $V = \{x, y : X,\ s : state\}$ and AX consist of the following Horn clauses:

$$\begin{aligned} sorted(\alpha) &\Leftarrow True \\ sorted(cons(x,\alpha)) &\Leftarrow True \\ sorted(cons(x, cons(y, s))) &\Leftarrow x \leq y \wedge sorted(cons(y, s)) \\ unsorted(cons(x, cons(y, s))) &\Leftarrow \neg(x \leq y) \\ unsorted(cons(x, cons(y, s))) &\Leftarrow unsorted(cons(y, s))) \end{aligned}$$

The least solution of AX in $Struct_{\Sigma,\mathcal{C}}$ interprets *sorted* and *unsorted* as the sets of sorted and unsorted lists over X, respectively.

The transformation of AX according to Theorem 11.3 (and exchanging predicate names) leads to the following co-Horn clauses whose greatest solution in $Struct_{\Sigma,\mathcal{C}}$ interprets *sorted* and *unsorted* also as the sets of sorted and unsorted lists, respectively:

$$\begin{aligned} unsorted(\alpha) &\Rightarrow False \\ unsorted(cons(x,\alpha)) &\Rightarrow False \\ unsorted(cons(x,cons(y,s))) &\Rightarrow \neg(x \leq y) \vee unsorted(cons(y,s))) \\ sorted(cons(x,cons(y,s))) &\Rightarrow x \leq y \\ sorted(cons(x,cons(y,s))) &\Rightarrow sorted(cons(y,s)) \end{aligned}$$

Example 4 (sequents for predicates on streams) Let X, $0 \in X$, $S = \{state\}$,

$$\begin{aligned} F &= \{head : state \to X,\ tail : state \to state,\ \ \leq: 1 \to \mathcal{P}(X \times X)\}, \\ \mathcal{C} &= X^{\mathbb{N}}, \qquad\qquad \text{(see sample algebra 9.6.5)} \\ P &= \{unsorted, has0 : 1 \to \mathcal{P}(state)\}, \end{aligned}$$

$\Sigma = (S, F \cup P)$, $V = \{s : state\}$ and AX consist of the following Horn clauses:

$$\begin{aligned} unsorted(s) &\Leftarrow \neg(head(s) \leq head(tail(s))) \vee unsorted(tail(s)) \\ has0(s) &\Leftarrow head(s) = 0 \vee has0(tail(s)) \end{aligned}$$

The least solution of AX in $Struct_{\Sigma,\mathcal{C}}$ interprets *unsorted* as the set of unsorted streams over X and $has0$ as the set of streams over X with at least one zero. Let

$$\begin{array}{rcl} SP &=& (\Sigma, AX, \mathcal{C}),\\ F' &=& F \cup \{has0 : 1 \to \mathcal{P}(state)\},\\ \mathcal{C}' &=& X^{\mathbb{N}},\ \forall\, f \in F : f^{\mathcal{C}'} = f^{\mathcal{C}},\ \ has0^{\mathcal{C}'} = has0^{lfp(\Phi_{SP})}\\ P' &=& \{sorted, not\_has0, has\infty 0, blink, blink' : 1 \to \mathcal{P}(state)\}, \end{array}$$

$\Sigma' = (S, F \cup P \cup P')$, $V = \{s : state\}$ and AX' consist of the following co-Horn clauses:

$$\begin{array}{rcl} sorted(s) &\Rightarrow& head(s) \leq head(tail(s)) \wedge sorted(tail(s))\\ not\_has0(s) &\Rightarrow& head(s) \neq 0 \vee not\_has0(tail(s))\\ has\infty 0(s) &\Rightarrow& has0(s) \wedge has\infty 0(tail(s))\\ blink(s) &\Rightarrow& head(s) = 0 \wedge blink'(s)\\ blink'(s) &\Rightarrow& head(s) = 1 \wedge blink(s) \end{array}$$

The greatest solution of AX' in $Struct_{\Sigma',\mathcal{C}'}$ interprets *sorted* as the set of sorted streams over X, not_has0 as the set of streams over X without zeros, $has\infty 0$ as the set of streams over X with infinitely many zeros and *blink* and $blink'$ as two sets of streams over X whose elements alternate between zero and nonzero components.

Let

$$\begin{aligned}
SP' &= (\Sigma', AX', \mathcal{C}'), \\
F'' &= F' \cup \{not_has0 : 1 \to \mathcal{P}(state)\}, \\
\mathcal{C}'' &= X^{\mathbb{N}},\ \forall\ f \in F' : f^{\mathcal{C}''} = f^{\mathcal{C}'},\ not_has0^{\mathcal{C}''} = not_has0^{gfp(\Phi_{SP'})} \\
P'' &= \{not_has\infty 0 : 1 \to \mathcal{P}(state)\},
\end{aligned}$$

$\Sigma'' = (S, F \cup P \cup P' \cup P'')$, $V = \{s : state\}$ and AX'' consist of the following Horn clause:

$$not\_has\infty 0(s) \ \Leftarrow\ not\_has0(s) \wedge not\_has\infty 0(tail(s))$$

The least solution of AX'' in $Struct_{\Sigma'',\mathcal{C}''}$ interprets $not\_has\infty 0$ as the set of streams over X with at most finitely many zeros. Again, the (co-)Horn axioms for the complement of a predicate p result from transforming the axioms for p (and exchanging predicate names) according to Theorem 11.3.

Example 5 The least solution of the Horn clause

$$sorted(s) \ \Leftarrow\ head(s) \leq head(tail(s)) \wedge sorted(tail(s))$$

is empty and thus not the set of sorted streams over X - as it might appear at first sight. Similarly, the greatest solution of the following co-Horn clause

$$unsorted(s) \ \Rightarrow\ \neg(head(s) \leq head(tail(s)) \vee unsorted(tail(s))$$

is the set of *all* streams over X and thus not the proper subset of unsorted streams.

Example 6 (Cartesian product and existential projection) Let $S = \{s, s'\}$,

$$\begin{aligned} P &= \{(*) : \mathcal{P}(s) \times \mathcal{P}(s') \to \mathcal{P}(s \times s'),\ \widehat{\exists} : \mathcal{P}(s \times s') \times \mathcal{P}(s') \to \mathcal{P}(s)\}, \\ \Sigma &= (S, F \cup P),\quad V = \{x, y : s,\ \varphi : \mathcal{P}(s),\ \psi : \mathcal{P}(s'),\ r : \mathcal{P}(s \times s')\} \end{aligned}$$

and AX consist of the following Horn clauses:

$$\begin{aligned} (\varphi * \psi)(x, y) &\Leftarrow \varphi(x) \wedge \psi(y) \\ \widehat{\exists}(r, \psi)(x) &\Leftarrow r(x, y) \wedge \psi(y) \end{aligned}$$

For all $\varphi : \mathcal{P}(s), \psi : \mathcal{P}(s'), r : \mathcal{P}(s \times s') \in \Lambda_{(S,F)}(V)$, the interpretations of $\varphi * \psi$ and $\widehat{\exists}(r, \psi)$ in $lfp(\Phi_{(\Sigma,AX,\mathcal{C})})$ coincides with $(\varphi * \psi)^{\mathcal{C}}$ and $\overline{\exists}(r)(\psi)^{\mathcal{C}}$, respectively (see sections 10.2 and 10.3).

Example 7 (Relational division and universal projection) Let $S = \{s, s'\}$,

$$\begin{aligned} P &= \{(/),\ \widehat{\forall} : \mathcal{P}(s \times s') \times \mathcal{P}(s') \to \mathcal{P}(s)\}, \\ \Sigma &= (S, F \cup P),\quad \{x, y : s,\ \psi : \mathcal{P}(s'),\ r : \mathcal{P}(s \times s')\} \end{aligned}$$

and AX consist of the following co-Horn clauses:

$$\begin{aligned} (r/\psi)(x) &\Rightarrow (\psi(y) \Rightarrow r(x, y)) \\ \widehat{\forall}(r, \psi)(x) &\Rightarrow (r(x, y) \Rightarrow \psi(y)) \end{aligned}$$

For all $\psi : \mathcal{P}(s'), r : \mathcal{P}(s \times s') \in \Lambda_{(S,F)}(V)$, the interpretations of r/ψ and $\widehat{\forall}(r, \psi)$ in $gfp(\Phi_{(\Sigma,AX,\mathcal{C})})$ coincides with $(r/\psi)^{\mathcal{C}}$ and $\overline{\forall}(r)(\psi)^{\mathcal{C}}$, respectively (see sections 10.2 and 10.3).

Example 8 (EF and AG) Let $S = \{state\}$, $(S, F) = KripkeSig$,

$$\begin{aligned} P &= \{EF : \mathcal{P}(state) \to \mathcal{P}(state)\}, \\ P' &= \{AG : \mathcal{P}(state) \to \mathcal{P}(state)\}, \\ \Sigma &= (S, F \cup P), \quad \Sigma' = (S, F \cup P'), \quad V = \{s, s' : state,\ \varphi : \mathcal{P}(state)\}, \end{aligned}$$

and AX and AX' consist of the following Horn and co-Horn clauses, respectively:

$$\begin{aligned} EF(\varphi)(s) &\Leftarrow \varphi(s) \\ EF(\varphi)(s) &\Leftarrow child(\emptyset)(s, s') \wedge EF(\varphi)(s') \\ AG(\varphi)(s) &\Rightarrow \varphi(s) \\ AG(\varphi)(s) &\Rightarrow (child(\emptyset)(s, s') \Rightarrow AG(\varphi)(s')) \end{aligned}$$

For all $\varphi : \mathcal{P}(state) \in \Lambda_{(S,F)}(V)$, the interpretation of $EF(\varphi)$ in $lfp(\Phi_{(\Sigma,AX,\mathcal{C})})$ coincides with $EF(\varphi)^{\mathcal{C}}$ and the interpretation of $AG(\varphi)$ in $gfp(\Phi_{(\Sigma',AX',\mathcal{C})})$ coincides with $AG(\varphi)^{\mathcal{C}}$ (see sections 10.2 and 10.3).

The examples show that Horn and co-Horn clauses yield the *formula* counterpart of the fixpoint operators, which were introduced as λ-*terms* in section 10.3. Hence induction and coinduction provide proof rules for sequents as they do for the corresponding μ- and ν-terms, respectively (see Theorem 10.8):

Theorem 11.4 (fixpoint induction and coinduction for sequents)

For all $p : e' \to \mathcal{P}(e) \in P$, let $\varphi_p : e' \to \mathcal{P}(e)$ be a closed (S, F)-λ-term.

(i) Let $SP = (\Sigma, AX, \mathcal{C})$ be a Horn specification of P such that for all $cl \in AX$, $\mathcal{A} = lfp(\Phi_{SP})$ satisfies $cl[\varphi_p/p \mid p \in P]$. Then for all $p : e' \to \mathcal{P}(e) \in P$, $x \in V_e$ and $z \in V_{e'}$, $\mathcal{A}$ satisfies the co-Horn clause

$$p(z)(x) \Rightarrow \varphi_p(z)(x),$$

i.e., $p(z)(x)^{\mathcal{A}} \subseteq \varphi(z)(x)^{\mathcal{A}}$.

(ii) Let $SP = (\Sigma, AX, \mathcal{C})$ be a co-Horn specification of P such that for all $cl \in AX$, $\mathcal{A} = lfp(\Phi_{SP})$ satisfies $cl[\varphi_p/p \mid p \in P]$. Then for all $p : e' \to \mathcal{P}(e) \in P$, $x \in V_e$ and $z \in V_{e'}$, $\mathcal{A}$ satisfies the Horn clause

$$p(z)(x) \Leftarrow \varphi_p(z)(x),$$

i.e., $\varphi(z)(x)^{\mathcal{A}} \subseteq p(z)(x)^{\mathcal{A}}$.

Proof. Let $\mathcal{B} \in Struct_{\Sigma,\mathcal{C}}$ be defined by $p^{\mathcal{B}} = \varphi_p^{\mathcal{A}}$ for all $p : e' \to \mathcal{P}(e) \in P$.

(i) By assumption, for all $cl \in AX$, $\mathcal{B}$ satisfies AX. Hence by Lemma 11.1 (5), $\mathcal{B}$ is Φ_{SP}-closed. Since $lfp(\Phi_{SP})$ is the least Φ_{SP}-closed Σ-algebra with (S,F)-reduct $\mathcal{C}$, for all $p : e' \to \mathcal{P}(e) \in P$ and $b \in A_{e'}$,

$$p^{\mathcal{A}}(b) \subseteq p^{\mathcal{B}}(b) = \varphi_p^{\mathcal{A}}(b) = \varphi_p^{\mathcal{C}}(b). \tag{1}$$

(i) is obtained by a sequence of equivalences: Let $x \in V_e$ and $z \in V_{e'}$.

$$\begin{aligned}
(1) \;&\Leftrightarrow\; \forall a \in A_e, b \in A_{e'} : (a \in p^{\mathcal{A}}(b) \Rightarrow a \in \varphi_p^{\mathcal{C}}(b)\\
&\Leftrightarrow\; \forall g \in A^V : (g(x) \in p^{\mathcal{A}}(g(z)) \Rightarrow g(x) \in \varphi_p^{\mathcal{C}}(g(z))\\
&\Leftrightarrow\; \forall g \in A^V : (g \in p(z)(x)^{\mathcal{A}} \Rightarrow g \in \varphi_p(z)(x)^{\mathcal{C}})\\
&\Leftrightarrow\; p(z)(x)^{\mathcal{A}} \subseteq \varphi_p(z)(x)^{\mathcal{C}}\\
&\Leftrightarrow\; A^V \setminus p(z)(x)^{\mathcal{A}} \cup \varphi_p(z)(x)^{\mathcal{C}} = A^V\\
&\Leftrightarrow\; \mathcal{A} \models \neg p(z)(x) \vee \varphi_p(z)(x)\\
&\Leftrightarrow\; \mathcal{A} \models p(z)(x) \Rightarrow \varphi_p(z)(x)
\end{aligned}$$

(ii) By assumption, for all $cl \in AX$, $\mathcal{B}$ satisfies AX. Hence by Lemma 11.2 (9), $\mathcal{B}$ is Φ_{SP}-dense.

Since $gfp(\Phi_{SP})$ is the greatest Φ_{SP}-dense Σ-algebra with (S,F)-reduct $\mathcal{C}$, for all $p : e' \to \mathcal{P}(e) \in P$ and $b \in A_{e'}$,

$$\varphi_p^{\mathcal{C}}(b) = \varphi_p^{\mathcal{A}}(b) = p^{\mathcal{B}}(b) \subseteq p^{\mathcal{A}}(b). \tag{2}$$

(ii) is obtained by a sequence of equivalences: Let $x \in V_e$ and $z \in V_{e'}$.

$$\begin{aligned}
(2) \;&\Leftrightarrow\; \forall a \in A_e, b \in A_{e'} : (a \in \varphi_p^{\mathcal{C}}(b) \Rightarrow a \in p^{\mathcal{A}}(b) \\
&\Leftrightarrow\; \forall g \in A^V : (g(x) \in \varphi_p^{\mathcal{C}}(g(z)) \Rightarrow g(x) \in p^{\mathcal{A}}(g(z))) \\
&\Leftrightarrow\; \forall g \in A^V : (g \in \varphi_p(z)(x)^{\mathcal{C}} \Rightarrow g \in p(z)(x)^{\mathcal{A}}) \\
&\Leftrightarrow\; \varphi_p(z)(x)^{\mathcal{C}} \subseteq p(z)(x)^{\mathcal{A}} \\
&\Leftrightarrow\; A^V \setminus \varphi_p(z)(x)^{\mathcal{C}} \cup p(z)(x)^{\mathcal{A}} = A^V \\
&\Leftrightarrow\; \mathcal{A} \models \neg\varphi_p(z)(x) \vee p(z)(x) \\
&\Leftrightarrow\; \mathcal{A} \models \varphi_p(z)(x) \Rightarrow p(z)(x)
\end{aligned}$$

❏

Rule-based versions of fixpoint induction and coinduction for sequents, which can even derive generalizations of the original conjectures, are presented in sections 12.1 and 13.1, respectively, and implemented in Expander2.

11.2 When Kleene closures are fixpoints

In chapters 12 and 13, proof rules for predicate specifications are presented that are based on the fixpoints of Φ_{SP} as derived from Theorem 3.9 (1) and (5) (see (3) and (4) above). If the Kleene closures of Φ_{SP} are Φ_{SP}-closed resp. -dense, the fixpoints are given by these closures (see Theorem 3.9 (4) and (8)).

The present section shows that Φ_{SP}^{∞} is Φ_{SP}-closed and $\Phi_{SP,\infty}$ is Φ_{SP}-dense if for all quantified subformulas $Qx\varphi$ of an axiom of SP, φ has only finitely many solutions in x:

An (S,F)-formula φ is **finitely $\mathcal{C}$-solvable** if for all $g \in A^V$,

$$Sol(\varphi, x, g) = \{a \in A_e \mid g[a/x] \in \varphi^{\mathcal{C}}\}$$

is finite.

$SP = (\Sigma, AX, \mathcal{C})$ is **finitely solvable** if

(i) AX consists of Horn clauses and for all subformulas $\forall x\varphi$ of the premise of a clause of AX, $\varphi \in Fo_{(S,F)}(V)$ or $\varphi = (\psi \Rightarrow \vartheta)$ for some finitely $\mathcal{C}$-solvable (S,F)-formula ψ and $\vartheta \in Fo_\Sigma(V)$, or

(ii) AX consists of co-Horn clauses and for all subformulas $\exists x\varphi$ of the conclusion of a clause of AX, $\varphi = (\psi \wedge \vartheta)$ for some finitely $\mathcal{C}$-solvable (S,F)-formula ψ and $\vartheta \in Fo_\Sigma(V)$.

Lemma 11.5

(i) Let $SP = (\Sigma, AX, \mathcal{C})$ be a finitely solvable Horn specification of P and $\Phi = \Phi_{SP}$. Then for all subformulas φ of the premise of a clause of AX,

$$\varphi^{\Phi^\infty} \subseteq \bigcup_{n<\omega} \varphi^{\Phi^n(\bot)}.$$

(ii) Let $SP = (\Sigma, AX, \mathcal{C})$ be a finitely solvable co-Horn specification of P and $\Phi = \Phi_{SP}$. Then for all subformulas φ of the conclusion of a clause of AX,

$$\bigcap_{n<\omega} \varphi^{\Phi^n(\top)} \subseteq \varphi^{\Phi_\infty}.$$

Proof of (i) by induction on the size of φ.

For all atoms $p(t) \in \Lambda_\Sigma(V)$ that are flat for P,

$$\begin{aligned} p(t)^{\Phi^\infty} &= \{g \in A^V \mid t^{\mathcal{C}}(g) \in p^{\Phi^\infty}(g)\} = \{g \in A^V \mid t^{\mathcal{C}}(g) \in p^{\bigsqcup_{n<\omega} \Phi^n(\bot)}(g)\} \\ &= \{g \in A^V \mid \exists n \in \mathbb{N} : t^{\mathcal{C}}(g) \in p^{\Phi^n(\bot)}(g)\} \\ &= \bigcup_{n<\omega} \{g \in A^V \mid t^{\mathcal{C}}(g) \in p^{\Phi^n(\bot)}(g)\} = \bigcup_{n<\omega} p(t)^{\Phi^n(\bot)}. \end{aligned}$$

For all $\varphi, \psi \in Fo_\Sigma(V)$ and $x : e \in V$,

$$\begin{aligned} (\varphi \vee \psi)^{\Phi^\infty} &= \varphi^{\Phi^\infty} \cup \psi^{\Phi^\infty} \overset{ind.\ hyp.}{\subseteq} (\textstyle\bigcup_{n<\omega} \varphi^{\Phi^n(\bot)}) \cup (\bigcup_{n<\omega} \psi^{\Phi^n(\bot)}) \\ &= \textstyle\bigcup_{n<\omega} (\varphi^{\Phi^n(\bot)} \cup \psi^{\Phi^n(\bot)}) = \bigcup_{n<\omega} (\varphi \vee \psi)^{\Phi^n(\bot)}, \\ (\varphi \wedge \psi)^{\Phi^\infty} &= \varphi^{\Phi^\infty} \cap \psi^{\Phi^\infty} \overset{ind.\ hyp.}{\subseteq} (\textstyle\bigcup_{n<\omega} \varphi^{\Phi^n(\bot)}) \cap (\bigcup_{n<\omega} \psi^{\Phi^n(\bot)}) \\ &= \textstyle\bigcup_{m,n<\omega} (\varphi^{\Phi^m(\bot)} \cap \psi^{\Phi^n(\bot)}) \subseteq \bigcup_{m,n<\omega} (\varphi^{\Phi^{max(m,n)}(\bot)} \cap \psi^{\Phi^{max(m,n)}(\bot)}) \\ &\subseteq \textstyle\bigcup_{n<\omega} (\varphi^{\Phi^n(\bot)} \cap \psi^{\Phi^n(\bot)}) = \bigcup_{n<\omega} (\varphi \wedge \psi)^{\Phi^n(\bot)}, \\ (\exists x \varphi)^{\Phi^\infty} &= \textstyle\bigcup_{a \in A_e} \{g \in A^V \mid g[a/x] \in \varphi^{\Phi^\infty}\} \\ &\overset{ind.\ hyp.}{\subseteq} \textstyle\bigcup_{a \in A_e} \{g \in A^V \mid g[a/x] \in \bigcup_{n<\omega} \varphi^{\Phi^n(\bot)}\} \\ &= \textstyle\bigcup_{n<\omega} \bigcup_{a \in A_e} \{g \in A^V \mid g[a/x] \in \varphi^{\Phi^n(\bot)}\} = \bigcup_{n<\omega} (\exists x \varphi)^{\Phi^n(\bot)}. \end{aligned}$$

Let $g \in A^V$. Suppose that there is $n \in \mathbb{N}$ such that

$$\forall\ a \in A_e : g[a/x] \in \varphi^{\Phi^\infty} \ \Rightarrow\ \forall\ a \in A_e : g[a/x] \in \varphi^{\Phi^n(\bot)}. \tag{15}$$

Then

$$\begin{aligned}(\forall x\varphi)^{\Phi^\infty} &= \textstyle\bigcap_{a\in A_e}\{g \in A^V \mid g[a/x] \in \varphi^{\Phi^\infty}\} \overset{(15)}{\subseteq} \bigcap_{a\in A_e}\{g \in A^V \mid g[a/x] \in \varphi^{\Phi^n(\bot)}\} \\ &= (\forall x\varphi)^{\Phi^n(\bot)} \subseteq \textstyle\bigcup_{n<\omega}(\forall x\varphi)^{\Phi^n(\bot)}.\end{aligned}$$

It remains to show (15): For all $a \in A_e$, let $g[a/x] \in \varphi^{\Phi^\infty}$. By induction hypothesis,

$$\forall\ a \in A_e : \exists\ n_a \in \mathbb{N} : g[a/x] \in \varphi^{\Phi^{n_a}(\bot)}. \tag{16}$$

Since SP is finitely solvable, $\varphi \in Fo_{(S,F)}(V)$ or there are $\psi \in Fo_{(S,F)}(V)$ and $\vartheta \in Fo_\Sigma(V)$ such that $\varphi = (\psi \Rightarrow \vartheta)$ and ψ is finitely $\mathcal{C}$-solvable. In the first case, for all $n \in \mathbb{N}$, $\varphi^{\Phi^\infty} = \varphi^{\mathcal{C}} = \varphi^{\Phi^n(\bot)}$, and thus (15) holds true trivially. In the second case, (16) implies

$$\forall\ a \in A_e : \exists\ n_a \in \mathbb{N} : (g[a/x] \in \psi^{\mathcal{C}} \Rightarrow g[a/x] \in \vartheta^{\Phi^{n_a}(\bot)}). \tag{17}$$

Since $Sol(\psi, x, g)$ is finite, $n = max\{n_a \mid g[a/x] \in \psi^{\mathcal{C}}\} < \omega$. Since $\vartheta^{\Phi^{n_a}(\bot)} \subseteq \vartheta^{\Phi^n(\bot)}$, (17) implies

$$\forall\ a \in A_e : (g[a/x] \in \psi^{\mathcal{C}} \Rightarrow g[a/x] \in \vartheta^{\Phi^n(\bot)})$$

and thus (15).

Proof of (ii) by induction on the size of φ.

For all atoms $p(t) \in \Lambda_\Sigma(V)$ that are flat for P,

$$\begin{aligned} p(t)^{\Phi\infty} &= \{g \in A^V \mid t^{\mathcal{C}}(g) \in p^{\Phi\infty}(g)\} = \{g \in A^V \mid t^{\mathcal{C}}(g) \in p^{\sqcap_{n<\omega}(\Phi^n(\top)}(g)\} \\ &= \{g \in A^V \mid \forall\, n \in \mathbb{N} : t^{\mathcal{C}}(g) \in p^{\Phi^n(\top)}(g)\} \\ &= \bigcap_{n<\omega}\{g \in A^V \mid t^{\mathcal{C}}(g) \in p^{\Phi^n(\top)}(g)\} = \bigcap_{n<\omega} p(t)^{\Phi^n(\top)}. \end{aligned}$$

For all $\varphi, \psi \in Fo_{\Sigma'}(V)$ and $x : e \in V$,

$$\begin{aligned} (\varphi \wedge \psi)^{\Phi\infty} &= \varphi^{\Phi\infty} \cap \psi^{\Phi\infty} \overset{ind.\ hyp.}{\supseteq} (\bigcap_{n<\omega} \varphi^{\Phi^n(\top)}) \cap (\bigcap_{n<\omega} \psi^{\Phi^n(\top)}) \\ &= \bigcap_{n<\omega}(\varphi^{\Phi^n(\top)} \cap \psi^{\Phi^n(\top)}) = \bigcap_{n<\omega}(\varphi \wedge \psi)^{\Phi^n(\top)}, \\ (\varphi \vee \psi)^{\Phi\infty} &= \varphi^{\Phi\infty} \cup \psi^{\Phi\infty} \overset{ind.\ hyp.}{\supseteq} (\bigcap_{n<\omega} \varphi^{\Phi^n(\top)}) \cup (\bigcap_{n<\omega} \psi^{\Phi^n(\top)}) \\ &= \bigcap_{m,n<\omega}(\varphi^{\Phi^m(\top)} \cup \varphi^{\Phi^n(\top)}) \supseteq \bigcap_{m,n<\omega}(\varphi^{\Phi^{min(m,n)}(\top)} \cup \varphi^{\Phi^{min(m,n)}(\top)}) \\ &\supseteq \bigcap_{n<\omega}(\varphi^{\Phi^n(\top)} \cup \psi^{\Phi^n(\top)}) = \bigcap_{n<\omega}(\varphi \vee \psi)^{\Phi^n(\top)}, \\ (\forall x\varphi)^{\Phi\infty} &= \bigcap_{a\in A_e}\{g \in A^V \mid g[a/x] \in \varphi^{\Phi\infty}\} \\ &\overset{ind.\ hyp.}{\supseteq} \bigcap_{a\in A_e}\{g \in A^V \mid g[a/x] \in \bigcap_{n<\omega} \varphi^{\Phi^n(\top)}\} \\ &= \bigcap_{n<\omega}\bigcap_{a\in A_e}\{g \in A^V \mid g[a/x] \in \varphi^{\Phi^n(\top)}\} = \bigcap_{n<\omega}(\forall x\varphi)^{\Phi^n(\top)}. \end{aligned}$$

Let $g \in \bigcap_{n<\omega}(\exists x\varphi)^{\Phi^n(\top)}$. Then for all $n \in \mathbb{N}$ there is $a_n \in A_e$ such that $g[a_n/x] \in \varphi^{\Phi^n(\top)}$. Since SP is finitely solvable, there are $\psi \in Fo_{(S,F)}(V)$ and $\vartheta \in Fo_\Sigma(V)$ such that $\varphi = (\psi \wedge \vartheta)$ and ψ is finitely $\mathcal{C}$-solvable. Hence for all $n \in \mathbb{N}$ there is $a_n \in A_e$ such that $g[a_n/x] \in \psi^{\mathcal{C}}$ and $g[a_n/x] \in \vartheta^{\Phi^n(\top)}$.

Since $Sol(\psi, x, g)$ is finite, there is $a \in A_e$ such that $a = a_n$ for infinitely many $n \in \mathbb{N}$. Hence for all $n \in \mathbb{N}$ there is $k_n \geq n$ such that $a = a_{k_n}$ and thus

$$g[a/x] = g[a_{k_n}/x] \in \psi^{\mathcal{C}} \cap \vartheta^{\Phi^{k_n}(\top)} = \varphi^{\Phi^{k_n}(\top)} \subseteq \varphi^{\Phi^n(\top)}.$$

Therefore, $g[a/x] \in \bigcap_{n<\omega}\varphi^{\Phi^n(\top)}$. By induction hypothesis, $g[a/x] \in \varphi^{\Phi_\infty}$ and thus $g \in (\exists x\varphi)^{\Phi_\infty}$. ❏

Theorem 11.6

(i) Let $SP = (\Sigma, AX, \mathcal{C})$ be a finitely solvable Horn specification of P. Then Φ^∞_{SP} is the least fixpoint of Φ_{SP}.

(ii) Let $SP = (\Sigma, AX, \mathcal{C})$ be a finitely solvable co-Horn specification of P. Then $\Phi_{SP,\infty}$ is the greatest fixpoint of Φ_{SP}.

Proof. Let $\Phi = \Phi_{SP}$.

(i) By Theorem 3.9 (4), it is sufficient to show that Φ^∞ is Φ-closed, i.e.,

$$\Phi(\Phi^\infty) \leq \Phi^\infty. \tag{18}$$

Let $p : e' \to \mathcal{P}(e) \in P$. Then for all $a \in A_e$ and $b \in A_{e'}$,

$$\begin{aligned}
& a \in p^{\Phi(\Phi^\infty)}(b) \\
\overset{(2)}{\Leftrightarrow}\ & \exists\, p(u)(t) \Leftarrow \varphi \in AX,\ g \in \varphi^{\Phi^\infty} : \langle t, u\rangle^{\mathcal{C}}(g) = (a, b) \\
\overset{\text{Lemma 11.5 (i)}}{\Rightarrow}\ & \exists\, p(u)(t) \Leftarrow \varphi \in AX,\ g \in \textstyle\bigcup_{n<\omega} \varphi^{\Phi^n(\bot)} : \langle t, u\rangle^{\mathcal{C}}(g) = (a, b) \\
\Leftrightarrow\ & \exists\, n < \omega,\ p(u)(t) \Leftarrow \varphi \in AX,\ g \in \varphi^{\Phi^n(\bot)} : \langle t, u\rangle^{\mathcal{C}}(g) = (a, b) \\
\overset{(2)}{\Leftrightarrow}\ & \exists\, n < \omega :\ a \in p^{\Phi^n(\bot)}(b) \\
\Leftrightarrow\ & a \in p^{\Phi^\infty}(b).
\end{aligned}$$

Hence (18) holds true.

(ii) By Theorem 3.9 (8), it is sufficient to show that Φ_∞ is Φ-dense, i.e.,

$$\Phi_\infty \leq \Phi(\Phi_\infty). \tag{19}$$

Let $p : e' \to \mathcal{P}(e) \in P$. Then for all $a \in A_e$ and $b \in A_{e'}$,

$$
\begin{aligned}
& a \in p_{\Phi_\infty}(b) \\
\Leftrightarrow\ & \forall\, n < \omega :\ a \in p^{\Phi^n(\top)}(b) \\
\overset{(2)}{\Leftrightarrow}\ & \forall\, n < \omega,\ p(u)(t) \Rightarrow \varphi \in AX,\ g \in A^V \setminus \varphi^{\Phi^n(\top)} : \langle t, u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
\Leftrightarrow\ & \forall\, n < \omega,\ p(u)(t) \Rightarrow \varphi \in AX,\ g \in A^V : g \in \varphi^{\Phi^n(\top)} \vee \langle t, u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
\Leftrightarrow\ & \forall\, p(t) \Rightarrow \varphi \in AX,\ g \in A^V : \forall\, n < \omega : g \in \varphi^{\Phi^n(\top)} \vee \langle t, u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
\Leftrightarrow\ & \forall\, p(t) \Rightarrow \varphi \in AX,\ g \in A^V : g \in \textstyle\bigcap_{n<\omega} \varphi^{\Phi^n(\top)} \vee \langle t, u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
\overset{Lemma\ 11.5\ (ii)}{\Rightarrow}\ & \forall\, p(t) \Rightarrow \varphi \in AX,\ g \in A^V : g \in \varphi^{\Phi_\infty} \vee \langle t, u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
\Leftrightarrow\ & \forall\, p(t) \Rightarrow \varphi \in AX,\ g \in A^V \setminus \varphi^{\Phi_\infty} : \langle t, u\rangle^{\mathcal{C}}(g) \neq (a,b) \\
\overset{(2)}{\Leftrightarrow}\ & a \in p^{\Phi_\infty}(b).
\end{aligned}
$$

Hence (19) holds true. ❑

Example 9 (EF and AG with quantifiers) Let Σ, Σ' and V be as in Example 8 and $SP = (\Sigma, AX, \mathcal{C})$ where AX consists of the following co-Horn clause:

$$EF(\varphi)(s) \ \Rightarrow\ \varphi(s) \vee \exists s' : (child(\emptyset)(s, s') \wedge EF(\varphi)(s'))$$

For all $\varphi : \mathcal{P}(state) \in \Lambda_{(S,F)}(V)$, the interpretation of $EF(\varphi)$ in $gfp(\Phi_{SP})$ coincides with $EF(\varphi)^{\mathcal{C}}$ (see sections 10.2 and 10.3).

Moreover, if $\mathcal{C}$ is finitely branching (see section 10.3), then SP is finitely solvable and thus by Theorem 11.6 (ii), $gfp(\Phi_{SP}) = \Phi_{SP,\infty}$.

Let $SP' = (\Sigma, AX', \mathcal{C})$ where AX' consists of the following Horn clause:

$$AG(\varphi)(s) \;\Leftarrow\; \varphi(s) \wedge \forall s' : (child(\emptyset)(s,s') \Rightarrow AG(\varphi)(s'))$$

For all $\varphi : \mathcal{P}(state) \in \Lambda_{(S,F)}(V)$, the interpretation of $AG(\varphi)$ in $lfp(\Phi_{SP'})$ coincides with $AG(\varphi)^{\mathcal{C}}$ (see sections 10.2 and 10.3). Moreover, if $\mathcal{C}$ is finitely branching, then SP' is finitely solvable and thus by Theorem 11.6 (i), $lfp(\Phi_{SP'}) = \Phi^{\infty}_{SP'}$.

11.3 Deduction in sequent logic

- **Top-down derivations** transform Σ-formulas into *True* or other formulas that represent solutions:

prove φ:	φ	$\vdash$	*True*
solve φ:	φ	$\vdash$	solved formula (see below)
refute φ:	$\neg\varphi$	$\vdash$	*True*
verify p:	$p(x) \Rightarrow \varphi$	$\vdash$	*True*
evaluate p:	$p(x) \Leftarrow \varphi$	$\vdash$	*True*
evaluate t:	$t = x$	$\vdash$	$x = u$
reduce t:	$t \rightarrow x$	$\vdash$	$\bigvee_{i=1}^{n} x = u_i$

A derivation

$$\varphi_1 \quad \vdash \quad \varphi_2 \quad \vdash \quad \dots \quad \vdash \quad \varphi_n$$

is sound with respect to the fixpoint semantics defined above, i.e., yields a sequence of reverse implications:

$$\varphi_1 \quad \Leftarrow \quad \varphi_2 \quad \Leftarrow \quad \dots \quad \Leftarrow \quad \varphi_n$$

The above goals are achieved if φ_1 and φ_n, respectively, look as follows:

prove φ_1: $\varphi_n = \mathit{True}$

refute φ_1: $\varphi_n = \mathit{False}$

solve φ_1: φ_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: $\varphi_1 = (t \to x)$, $\varphi_n = (x = u_1) \vee \cdots \vee (x = u_k)$

Other derivations performed by Expander2 are sequences of (sets of) Σ-formulas of an arbitrary type:

$$t_1 \;\vdash\; t_{21}{<}{+}{>}\dots{<}{+}{>}t_{2k_2} \;\vdash\; \dots \;\vdash\; t_{n1}{<}{+}{>}\dots{<}{+}{>}t_{nk_n}$$

<+> 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 chapters 12 and 13). They apply **goals** (hypotheses) to **axioms** and thus prove the former by solving the latter.

11.4 Rule applicability

Let $\mathcal{A}$ be a Σ'-algebra and $\varphi, C(\varphi), \psi$ be Σ'-formulas such that φ is a subformula of $C(\varphi)$.

- $\dfrac{\varphi}{\psi}\Updownarrow$ 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:
  $$\frac{C[\varphi]}{C[\psi]}\Updownarrow$$
- $\dfrac{\varphi}{\psi}\Uparrow$ denotes an **expansion rule for $\mathcal{A}$**, i.e., $\mathcal{A}$ satisfies $\psi \Rightarrow \varphi$.
  Applied in a context $C[\varphi]$ where φ has positive polarity, it leads to a further expansion rule:
  $$polarity(position(\varphi), C[\varphi]) = + \quad \Longrightarrow \quad \frac{C[\varphi]}{C[\psi]}\Uparrow$$
- $\dfrac{\varphi}{\psi}\Downarrow$ denotes a **contraction rule for $\mathcal{A}$**, i.e., $\mathcal{A}$ satisfies $\varphi \Rightarrow \psi$.
  Applied in a contecxt $C[\varphi]$ where φ has negative polarity, it leads to an expansion rule:
  $$polarity(position(\varphi), C[\varphi]) = - \quad \Longrightarrow \quad \frac{C[\varphi]}{C[\psi]}\Uparrow$$

11.5 Resolution and narrowing

(see also [129, 138, 131])

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 appications.

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** [131, 138] 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 predicate**

Let $\gamma_1 \Rightarrow (p(t_1) \Longleftarrow \varphi_1),\dots,\gamma_n \Rightarrow (p(t_n) \Longleftarrow \varphi_n)$ 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.

$$\frac{p(t)}{\bigvee_{i=1}^{k} \exists Z_i : (\varphi_i\sigma_i \wedge \vec{x} = \vec{x}\sigma_i)} \quad \Updownarrow$$

If only a single axiom for p is applied, then the corresponding rule is only an expansion rule.

- **Narrowing upon a function**

Let $\gamma_1 \Rightarrow (f(t_1) = u_1 \Longleftarrow \varphi_1),\dots,\gamma_n \Rightarrow (f(t_n) = u_n \Longleftarrow \varphi_n)$ be all Horn clauses for f in AX, $at(x)$ be a Σ'-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 $k < i \leq l$, σ_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**

  Let $\gamma_1 \Rightarrow (t_1 \to u_1 \Longleftarrow \varphi_1), \ldots, \gamma_n \Rightarrow (t_n \to u_n \Longleftarrow \varphi_n)$ be all Horn clauses for $\to$ in AX, σ_i be a unifier modulo associativity and commutativity of $^\wedge$ and $(**)$ hold true.

$$\frac{t\,^\wedge v \to t'}{\bigvee_{i=1}^{k} \exists Z_i : ((u_i{}^\wedge v)\sigma_i = t'\sigma_i \wedge \varphi_i\sigma_i \wedge \vec{x} = \vec{x}\sigma_i) \vee \\ \bigvee_{i=k+1}^{l} ((t\,^\wedge v)\sigma_i \to t'\sigma_i \wedge \vec{x} = \vec{x}\sigma_i)} \Updownarrow$$

  Again, if only a single axiom for $\to$ is applied, then the corresponding rule is only an expansion rule.

As pointed out in [14, 120, 121], 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**

  Let $f(t_1) = u_1 \Longleftarrow \varphi_1, \dots, f(t_n) = u_n \Longleftarrow \varphi_n$ 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.

$$\frac{f(t)}{u_1\sigma_1 <+> \cdots <+> u_k\sigma_k}$$

- **Rewriting upon a transition relation**

  Let $t_1 \to u_1 \Longleftarrow \varphi_1, \dots, t_n \to u_n \Longleftarrow \varphi_n$ be all Horn clauses for $\to$ in AX and $(***)$ hold true.

$$\frac{t}{u_1\sigma_1 <+> \cdots <+> \cdots <+> \cdots <+> u_k\sigma_k}$$

- **Coresolution upon a predicate p**

  Let $AX_p = \{\gamma_1 \Rightarrow (p(t_1) \Longrightarrow \varphi_1), \dots, \gamma_n \Rightarrow (p(t_n) \Longrightarrow \varphi_n)\}$ 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 Σ'-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.

$$\frac{p(t)}{False} \qquad \frac{q(t)}{True} \qquad \frac{at(f(t))}{at()} \qquad \frac{t \rightarrow t'}{() \rightarrow t'} \quad \Updownarrow$$

  The elimination rules are correct only if p, q, f and $\rightarrow$ are axiomatized completely.

12 Induction rules

Let $\Sigma = (S, F \cup \{p : e' \to \mathcal{P}(e)\})$ be a signature, $\mathcal{C}$ be an (S,F)-algebra and $SP = (\Sigma, AX, \mathcal{C})$ be a Horn specification of P. For simplicity, we restrict ourselves to a single predicate p. The generalization to several predicates is straightforward (see Theorem 11.4 (1)).

12.1 Fixpoint induction upon a predicate

Let $p : e' \to \mathcal{P}(e) \in P$, φ be a closed (S,F)-λ-term, $x \in V_e$ and $z \in V_{e'}$.

A proof by fixpoint induction that $\mathcal{A} = lfp(\Phi_{SP})$ satisfies $p(x) \Rightarrow \varphi$ is a sequence $(\psi_1, \ldots, \psi_n)$ of Σ-formulas such that the following conditions hold true:

- ψ_2 is the result of applying the following rule to ψ_1:

$$(1) \qquad \frac{p(z)(x) \Rightarrow \varphi}{\bigwedge_{p(u)(t) \Leftarrow \delta \in AX} (\delta[q/p] \Rightarrow \varphi[t/x, u/z])} \Uparrow$$

  After applying (1), the predicate $q : e' \to \mathcal{P}(e)$ and the co-Horn clause $q(z)(x) \Rightarrow \varphi$ are added to SP.

- For all $1 < i < n$, ψ_{i+1} is the result of applying to ψ_i an expansion rule for $\mathcal{C}$ (see section 11.4) or the following rule:

$$(2) \qquad \frac{q(z)(x) \Rightarrow \varphi'}{\bigwedge_{p(u)(t) \Leftarrow \delta \in AX}(\delta[q/p] \Rightarrow \varphi'[t/x, u/z])}$$

  After applying (2), the co-Horn clause $q(z)(x) \Rightarrow \varphi'$ is added to SP.

- $\psi_n = True$.

(1) is an expansion rule for $\mathcal{A} = lfp(\Phi_{SP})$: If the succedent of (1) holds true in $\mathcal{A}$, then $\mathcal{A}$ satisfies the axioms for p if p were replaced by φ. Since $\mathcal{A}$ interprets p as the *least* relation satisfying the axioms for p, we conclude that the antecedent of (1) holds true in $\mathcal{A}$.

Proof sketch of the correctness of $(\psi_1, \ldots, \psi_n)$

Suppose that the derivation $(\psi_1, \ldots, \psi_n)$ contains k applications of (2). Then it reads schematically as follows:

$$
\begin{array}{ll}
 & p(z)(x) \Rightarrow \varphi \\
\stackrel{(1)}{\vdash} & \bigwedge_{p(u)(t)\Leftarrow\delta\in AX}(\delta[q/p] \Rightarrow \varphi[t/x,u/z]) \qquad (*) \\
\stackrel{expansion\ rules}{\vdash} & \dots\ q(z)(x) \Rightarrow \varphi_1\ \dots \\
\stackrel{(2)}{\vdash} & \dots\ \bigwedge_{p(u)(t)\Leftarrow\delta\in AX}(\delta[q/p] \Rightarrow \varphi_1[t/x,u/z])\ \dots \\
\vdash & \dots \\
\stackrel{expansion\ rules}{\vdash} & \dots\ q(z)(x) \Rightarrow \varphi_k\ \dots \\
\stackrel{(2)}{\vdash} & \dots\ \bigwedge_{p(u)(t)\Leftarrow\delta\in AX}(\delta[q/p] \Rightarrow \varphi_k[t/x,u/z])\ \dots \\
\stackrel{expansion\ rules}{\vdash} & True
\end{array}
$$

Since $q \not\in \varphi \wedge \varphi_1 \wedge \cdots \wedge \varphi_k$, $q(z)(x)$ is equivalent to φ before the first application of (2), while—due to the stepwise addition of axioms for q (see above)—for all $1 \leq i \leq k$, $q(z)(x)$ is equivalent to $\varphi \wedge \varphi_1 \wedge \cdots \wedge \varphi_i$ 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, $q(z)(x)$ is considered to be equivalent to $\varphi \wedge \varphi_1 \wedge \cdots \wedge \varphi_k$.

Then for all $1 \leq i \leq k$, $q(z)(x) \Rightarrow \varphi_i$ holds true, and thus the subderivation starting with $(*)$ yields the validity of

$$\bigwedge_{p(t)\Leftarrow\delta\in AX} (\delta[q/p] \Rightarrow \varphi[t/x, u/z]) \tag{3}$$

and

$$\bigwedge_{p(u)(t)\Leftarrow\delta\in AX} \bigwedge_{i=1}^{k} (\delta[q/p] \Rightarrow \varphi_i[t/x, u/z]). \tag{4}$$

(3)∧(4) is equivalent to

$$\bigwedge_{p(u)(t)\Leftarrow\delta\in AX} (\delta[q/p] \Rightarrow (\varphi \wedge \varphi_1 \wedge \cdots \wedge \varphi_k)[t/x, u/z])$$

and thus to $\bigwedge_{p(u)(t)\Leftarrow\delta\in AX}(\delta[q/p] \Rightarrow q[t/x, u/z])$. Hence q (instead of p) satisfies AX in $\mathcal{A}$ and thus by the correctness of (1),

$$p(z)(x) \Rightarrow q(x) \tag{5}$$

and, in particular, the original goal $p(z)(x) \Rightarrow \varphi$ hold true in $\mathcal{A}$.

$q(z)(x)$ can be regarded as a **generalization** of φ. By (5), $q(z)(x)$ lies *somewhere* between $p(z)(x)$ and φ, the least (!) relation satisfying AX:

$$p(z)(x) \;\Rightarrow\; q(z)(x) \;\Rightarrow\; \varphi.$$

Therefore, the validity of an inductive conjecture like $p(z)(x) \Rightarrow \varphi$ is not semi-decidable, let alone decidable.

If p were a *greatest* predicate, then proving conjectures of the form $p(z)(x) \Rightarrow \varphi$ amounts to coresolving them upon p (see sections 11.5 and 13.6).

12.2 Invariants and algebraic induction

Let $\Sigma = (S, F)$ be a signature, $\mathcal{A}$ be a Σ-algebra with carrier A and B be a Σ-invariant of $\mathcal{A}$.

$inc_B : \mathcal{A}|_B \to \mathcal{A}$ denotes the Σ-homomorphic inclusion map (see section 9.1).

Let $h : A \to B$ 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 12.1 (Homomorphisms and invariants)

(1) Let $h : A \to B$ be an S-sorted function and $\mathcal{B}$ be a Σ-algebra with carrier B. A can be extended to a Σ-algebra $\mathcal{A}$ and h to a Σ-homomorphism from $\mathcal{A}$ to $\mathcal{B}$ iff $img(h)$ is a Σ-invariant.

(2) $h : \mathcal{A} \to \mathcal{B}$ is Σ-homomorphic iff there is a unique Σ-epimorphism $h' : \mathcal{A} \to \mathcal{B}|_{img(h)}$ with $inc_{img(h)} \circ h' = h$. Hence, if h is mono, then by Lemma 4.1 (2), h' is mono and thus $\mathcal{A}$ and $\mathcal{B}|_{img(h)}$ are Σ-isomorphic.

Proof. (1) If h is Σ-homomorphic, then $img(h)$ is a Σ-invariant. Let $img(h)$ be a Σ-invariant. For all $f : e \to e' \in F$, define $f^{\mathcal{A}} : A_e \to A_{e'}$ 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 Σ-algebra and h is Σ-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 9.1 (2). ❑

Moreover, by Theorem 3.4 (1), for every S-sorted subset B of A, the least Σ-invariant $\langle B\rangle$ containing B is the union of all B_n, $n\in\mathbb{N}$, with $B_0=B$ and for all $s\in S$,

$$B_{n+1,s}=\{c^A(a)\mid c:e\to s\in C,\ a\in B_{n,e}\}.$$

Proofs by algebraic induction

Let $C\subseteq F$ be a set of constructors and $C\Sigma=(S,C)$.

❀ A Σ-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 Σ-algebra with carrier A that satisfies the induction principle for C and for all $s\in S$, φ_s be a Σ-formula such that $free(\varphi_s)\subseteq\{x\}\subseteq V_s$.

Then the (in)validity of φ_s, $s\in S$, in $\mathcal{A}$ may be proved by the following iterative algorithm:

- **Step 1**: For all $s\in S$, set $B_s:=B_{0,s}=_{def}\{g(x)\mid g\in\varphi_s^{\mathcal{A}}\}$.
- **Step 2**: For all $s\in S$, let $B'_s=\{c^{\mathcal{A}}(a)\mid c:e\to s\in C,\ a\in B_e\}$.

- **Step 3**: If $B' \subseteq B$, then stop: B is a $C\Sigma$-invariant of $\mathcal{A}$ 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 φ.
  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 12.2

Let $h : \mathcal{A} \to \mathcal{B}$ be Σ-homomorphic, A be the carrier of $\mathcal{A}$ and inv be a Σ-invariant of $\mathcal{B}$.

$$h^{-1}(inv) \;=\; \{a \in A \mid h(a) \in inv\}$$

is a Σ-invariant of $\mathcal{A}$.

Proof. Let $f : e \to e' \in F$, $a \in A_e$ 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 Σ-homomorphic and inv is a Σ-invariant. Hence $f^{\mathcal{A}}(a) \in h^{-1}(inv)$. ❏

Lemma 12.3 (Induction and initiality)

Let $\Sigma = (S, F)$ be a constructive signature and $\mathcal{A}, \mathcal{B}$ be Σ-algebras with carriers A, B and $\mathcal{K}$ be a full subcategory of Alg_Σ that is closed under subalgebras.

(1) $\mathcal{A}$ satisfies the induction principle iff A is the only Σ-invariant of $\mathcal{A}$.

(2) If A is the only Σ-invariant of $\mathcal{A}$, then all Σ-homomorphisms from $\mathcal{A}$ to $\mathcal{B}$ coincide.

(3) $\mathcal{A}$ is initial in $\mathcal{K}$ iff A is the only Σ-invariant of $\mathcal{A}$ and for all $\mathcal{B} \in \mathcal{K}$ there is a Σ-homomorphism from $\mathcal{A}$ to $\mathcal{B}$.

(4) If $\mathcal{A}$ is initial in $\mathcal{K}$, then the image of the unique Σ-homomorphism $fold^{\mathcal{B}} : \mathcal{A} \to \mathcal{B}$ is the least Σ-invariant of $\mathcal{B}$.

Proof.

(1) "$\Rightarrow$": Every Σ-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 Σ-invariant and A is the only Σ-invariant of $\mathcal{A}$. Then $A \subseteq B$.

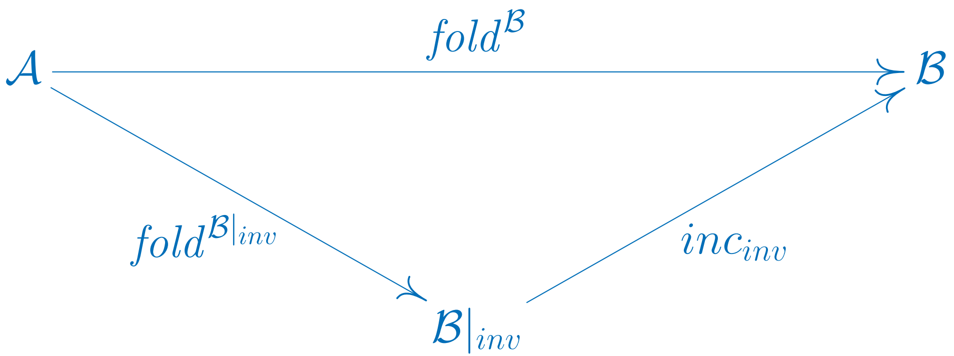
(2) Let $g, h : \mathcal{A} \to \mathcal{B}$ be Σ-homomorphisms. Then $B = \{a \in A \mid g(a) = h(a)\}$ is a Σ-invariant of $\mathcal{A}$: Let $f : e \to e' \in F$ and $a \in A_e$.

Since g and h are Σ-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 7.2 (3) implies $f^{\mathcal{A}}(a) \in B_{e'}$. Since A is the only Σ-invariant of $\mathcal{A}$, B agrees with A and thus for all $a \in A$, $g(a) = h(a)$. Hence $g = h$.

(3) "⇒": Let $\mathcal{A}$ be initial in $\mathcal{K}$ and B be a Σ-invariant of A. B induces a Σ-monomorphism $inc_B : \mathcal{A}|_B \to \mathcal{A}$. Hence Lemma 4.3 (1) implies that inc_B is iso in $\mathcal{K}$ and thus $B = A$. Since $\mathcal{A}$ is initial in $\mathcal{K}$, there is a Σ-homomorphism from $\mathcal{A}$ to $\mathcal{B}$.

"⇐": Suppose that A is the only Σ-invariant of $\mathcal{A}$ and for all $\mathcal{B} \in \mathcal{K}$ there is a Σ-homomorphism $h : \mathcal{A} \to \mathcal{B}$. By (2), h is unique. Hence $\mathcal{A}$ is initial in $\mathcal{K}$.

(4) Let inv be a Σ-invariant of $\mathcal{B}$. Since $\mathcal{A}$ is initial in $\mathcal{K}$, the following diagram commutes:



Hence for all $a \in A$,

$$fold^{inv}B(a) = inc_{inv}(fold^{\mathcal{B}|_{inv}}(a)) = fold^{\mathcal{B}|_{inv}}(a) \in inv.$$

We conclude that inv contains $img(fold^{\mathcal{B}})$.

Alternative proof of (4): By Lemma 12.2,

$$(fold^{\mathcal{B}})^{-1}(inv) \;=\; \{a \in A \mid fold^{\mathcal{B}}(a) \in inv\}$$

is a Σ-invariant of $\mathcal{A}$. By (3), A is the only Σ-invariant of $\mathcal{A}$. Hence $(fold^{B})^{-1}(inv) = A$ and thus for all $b \in inv$ there is $a \in A$ with $fold^{\mathcal{B}}(a) = b$, i.e., $b \in img(fold^{\mathcal{B}})$. ❑

12.3 CFGs as equations between regular expressions

Let $G = (S, X, R)$ be a context-free grammar (see section 9.14).

R induces the set E_G of $Reg(X)$-equations over S (see section 9.11):

$$E_G \;=\; \{s = \overline{w_1} + \cdots + \overline{w_n} \mid s \in S,\ \{w_1, \ldots, w_n\} = \{w \in S_X^* \mid s \to w \in R\}\}$$

where $\overline{\epsilon} = \widehat{1}$, for all $s_1, \ldots, s_n \in S_X$, $\overline{s_1 \ldots s_n} = \overline{s_1} * \cdots * \overline{s_n}$, and for all $s \in S_X$,

$$\overline{s} \;=\; \begin{cases} s & \text{if } s \in S, \\ \overline{s} & \text{otherwise.} \end{cases}$$

Example 12.4 The rules of $(S, X, R) = \text{SAB}$ (see Example 9.10) yield the following set E_G of $Reg(X)$-equations over S:

$$C = \overline{a} * B + \overline{b} * A,$$

$$A = \overline{a} * C + \overline{b} * A * A,$$

$$B = \overline{b} * C + \overline{a} * B * B.$$

Theorem 12.5 (The language of a CFG solves its equations)

(i) The valuation $g = \lambda s.L(G)_s \in L(G)^S$ solves E_G in $Lang(X)$ (see sample algebra 9.6.19).

Let $g \in L(G)^S$ solve E_G in $Lang(X)$.

(ii) The S-sorted set A with $A_s = g(s)$ for all $s \in S$ is the carrier of a $\Sigma(G)$-subalgebra of $Word(G)$.

(iii) $\lambda s.L(G)_s$ is the least solution (w.r.t. the inclusion of its carriers) E_G in $Lang(X)$.

Proof. Let $s = \overline{w_1} + \cdots + \overline{w_n} \in E_G$ and $1 \le i \le n$.

Then $s \to w_i \in R$ and $w_i = v_0 s_{i,1} v_1 \dots s_{i,n_i} v_{n_i}$ for some $v_0 \dots v_{n_i} \in X^*$ and $s_{i,1}, \dots, s_{i,n_i} \in S_X$.

Hence $src(f_{s \to w_i}) = s_{i,1} \times \dots \times s_{i,n_i}$ and

$$\begin{aligned} & g^*(\overline{w_i}) = g^*(\overline{v_0 s_{i,1} v_1 \dots s_{i,n_i} v_{n_i}}) = g^*(\overline{v_0} * \overline{s_{i,1}} * \overline{v_1} * \cdots * \overline{s_{i,n_i}} * \overline{v_{n_i}}) \\ & = g^*(\overline{v_0}) \cdot g^*(\overline{s_{i,1}}) \cdot g^*(\overline{v_1}) \cdot \cdots \cdot g^*(\overline{s_{i,n_i}}) \cdot g^*(\overline{v_{n_i}}) = v_0 \cdot L(G)_{s_{i,1}} \cdot v_1 \cdot \cdots \cdot L(G)_{s_{i,n_i}} \cdot v_{n_i} \\ & = f_{s \to w_i}^{Word(G)}(L(G)_{s_{i,1}}, \dots, L(G)_{s_{i,n_i}}). \end{aligned} \tag{1}$$

Proof of (i).

$$\begin{aligned} & g(s) = L(G)_s = fold_s^{Word(G)}(T_{\Sigma(G),s}) \\ & = \bigcup_{i=1}^n \{fold_s^{Word(G)}(f_{s \to w_i}(t)) \mid t \in T_{\Sigma(G), s_{i,1} \times \dots \times s_{i,n_i}}\} \\ & = \bigcup_{i=1}^n \{f_{s \to w_i}^{Word(G)}(fold_{s_{i,1} \times \dots \times s_{i,n_i}}^{Word(G)}(t)) \mid t \in T_{\Sigma(G), s_{i,1} \times \dots \times s_{i,n_i}}\} \\ & = \{f_{s \to w_i}^{Word(G)}(v) \mid v \in L(G)_{s_{i,1} \times \dots \times s_{i,n_i}}, 1 \le i \le n_i\} \\ & = \bigcup_{i=1}^n f_{s \to w_i}^{Word(G)}(L(G)_{s_{i,1}}, \dots, L(G)_{s_{i,n_i}}) \overset{(1)}{=} g^*(\overline{w_1}) \cup \cdots \cup g^*(\overline{w_n}) \\ & = g^*(\overline{w_1} + \cdots + \overline{w_n}), \end{aligned}$$

i.e., g solves $s = \overline{w_1} + \cdots + \overline{w_n}$ in $Lang(X)$.

Proof of (ii).

Suppose that for all $1 \leq i \leq n$,

$$f_{s\to w_i}^{Word(G)}(A_{s_{i,1}},\ldots,A_{s_{i,n_i}}) \;\subseteq\; A_s. \tag{2}$$

Then (ii) holds true. It remains to show (2). Since g solves E_G in $Lang(X)$,

$$g(s) = g^*(\overline{w_1}+\cdots+\overline{w_n}) = g^*(\overline{w_1})\cup\cdots\cup g^*(\overline{w_n}) \supseteq g^*(\overline{w_i}). \tag{3}$$

Hence for all $1 \leq i \leq n$,

$$\begin{aligned}
&f_{s\to w_i}^{Word(G)}(A_{s_{i,1}},\ldots,A_{s_{i,n_i}}) = v_0\cdot A_{s_{i,1}}\cdot v_1\cdot\cdots\cdot A_{s_{i,n_i}}\cdot v_{n_i}\\
&= g^*(\overline{v_0})\cdot g(s_{i,1})\cdot g^*(\overline{v_1})\cdot\cdots\cdot\; g(s_{i,n_i})\cdot g^*(\overline{v_{n_i}}) = g^*(\overline{v_0}*\overline{s_{i,1}}*\overline{v_1}*\cdots*\overline{s_{i,n_i}}*\overline{v_{n_i}})\\
&= g^*(\overline{v_0 s_{i,1} v_1\ldots s_{i,n_i} v_{n_i}}) = g^*(\overline{w_i}) \overset{(3)}{\subseteq} g(s) = A_s,
\end{aligned}$$

i.e., (2) holds true.

Proof of (iii).

By (i), $\lambda s.L(G)_s$ solves E_G. By Lemma 12.3 (4), $fold^{Word(G)}(T_{\Sigma(G)})$ is the least $\Sigma(G)$-subalgebra of $Word(G)$. Hence for all $s \in S$,

$$L(G)_s = fold_s^{Word(G)}(T_{\Sigma(G),s}) \overset{(ii)}{\subseteq} A_s = g(s). \qquad ❏$$

Theorem 12.6 (see [87], pp. 53 f.)

Let $G = (S, X, R)$, $\mathcal{A} = Lang(X)$, $s \in S$, $t_1, \ldots, t_n, t \in T_{Reg(X)}$ such that for some $s \in S$,

$$s = t_1 * s + \cdots + t_n * s + t \tag{1}$$

is the only equation of E_G with left-hand side s and for all $1 \leq i \leq n$, $\epsilon \notin t_i^{\mathcal{A}}$.

$g \in L(G)^S$ with $g(s) = (star(t_1 + \cdots + t_n) * t)^{\mathcal{A}}$ is the unique solution of (1) in $\mathcal{A}$.

Proof. Since $h \in L(G)^S$ solves (1) in $\mathcal{A}$ iff h solves $s = (t_1 + \cdots + t_n) * s + t$ in $\mathcal{A}$, it suffices to show that for all $t, u \in T_{Reg(X)}$, $g \in L(G)^S$ with $g(s) = (star(t) * u)^{\mathcal{A}}$ and $\epsilon \notin t^{\mathcal{A}}$ is the unique solution of

$$s = t * s + u \tag{2}$$

in $\mathcal{A}$.

g solves (2) in $\mathcal{A}$:

$$g(s) = (star(t) * u)^{\mathcal{A}} = (t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} = t^{\mathcal{A}} \cdot (t^{\mathcal{A}})^* \cdot u \cup u^{\mathcal{A}} = t^{\mathcal{A}} \cdot (t^{\mathcal{A}})^* \cdot u \cup u^{\mathcal{A}}$$
$$= t^{\mathcal{A}} \cdot (star(t) * u)^{\mathcal{A}} \cup u^{\mathcal{A}} = t^{\mathcal{A}} \cdot g(s) \cup u^{\mathcal{A}} = g^*(t * s + u).$$

g is the least solution of (2) in $\mathcal{A}$:

Since $g(s) = (t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} = \bigcup_{i \in \mathbb{N}} (t^{\mathcal{A}})^i \cdot u^{\mathcal{A}}$, g is the least solution of (2) in $\mathcal{A}$ if for all solutions $h \in L(G)^S$ of (2) in $\mathcal{A}$ and $i \in \mathbb{N}$,

$$\bigcup_{j=0}^{i} (t^{\mathcal{A}})^j \cdot u^{\mathcal{A}} \;\subseteq\; h(s). \tag{3}$$

Proof of (3) by induction on i. Since h solves (2) in $\mathcal{A}$,

$$t^{\mathcal{A}} \cdot h(s) \cup u^{\mathcal{A}} = h^*(t * s + u) = h(s). \tag{4}$$

Hence we obtain (3) for $i = 0$:

$$(t^{\mathcal{A}})^0 \cdot u^{\mathcal{A}} = \{\epsilon\} \cdot u^{\mathcal{A}} = u^{\mathcal{A}} = t^{\mathcal{A}} \cdot \emptyset \cup u^{\mathcal{A}} \subseteq t^{\mathcal{A}} \cdot h(s) \cup u^{\mathcal{A}} \overset{(4)}{=} h(s).$$

By induction hypothesis,

$$\bigcup_{j=0}^{i-1} (t^{\mathcal{A}})^j \cdot u^{\mathcal{A}} \subseteq h(s). \tag{5}$$

Hence we obtain (3) for $i > 0$:

$$\bigcup_{j=0}^{i} (t^{\mathcal{A}})^j \cdot u^{\mathcal{A}} = \bigcup_{j=0}^{i-1} (t^{\mathcal{A}})^{j+1} \cdot u^{\mathcal{A}} \cup u^{\mathcal{A}} = t^{\mathcal{A}} \cdot \bigcup_{j=0}^{i-1} (t^{\mathcal{A}})^j \cdot u^{\mathcal{A}} \cup u^{\mathcal{A}}$$
$$\overset{(5)}{\subseteq} t^{\mathcal{A}} \cdot h(s) \cup u^{\mathcal{A}} \overset{(4)}{=} h(s).$$

g is the only solution of (2) in $\mathcal{A}$: Let $h \in L(G)^S$ be a solution of (2) in $\mathcal{A}$. Since $(t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} = g(s) \subseteq h(s)$, there is $L \subseteq X^*$ with $h(s) = (t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} \cup L$ and $L \cap (t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} = \emptyset$.

Since h solves (2),

$$\begin{aligned}(t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} \cup L = h(s) = t^{\mathcal{A}} \cdot h(s) \cup u^{\mathcal{A}} = t^{\mathcal{A}} \cdot ((t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} \cup L) \cup u^{\mathcal{A}} \\ = t^{\mathcal{A}} \cdot (t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} \cup t^{\mathcal{A}} \cdot L \cup u^{\mathcal{A}} = (t^{\mathcal{A}})^* \cdot u^{\mathcal{A}} \cup t^{\mathcal{A}} \cdot L\end{aligned} \tag{6}$$

Let $w \in L$ be of minimal length and $|w| = k$. Since $w \notin (t^{\mathcal{A}})^* \cdot u^{\mathcal{A}}$, (6) implies $w \in t^{\mathcal{A}} \cdot L$. Since $\epsilon \notin t^{\mathcal{A}}$, for all $v \in t^{\mathcal{A}} \cdot L$, $|v| > k$. Hence $w \notin t^{\mathcal{A}} \cdot L$. $\lightning$ ❏

When turning the transition graph represented by an $Acc(X)$-algebra into $Reg(X)$-equations, one often encounters equations of the form (1). Since $\mathcal{A}$ solves them uniquely, they can then be replaced by their respective solutions (see [138], chapter 8). ❏

12.4 Algebraic induction as fixpoint induction

Let $C \subseteq F$ be a set of constructors, $C\Sigma = (S, C)$, $\mathcal{C}$ be an initial $C\Sigma$-algebra and for all $s \in S$, $\varphi_s \in Fo_\Sigma(V)$ such that $free(\varphi_s) = \{x\} \subseteq V_s$.

Moreover, let $P = \{inv_e : \mathcal{P}(e) \mid e \in \mathcal{T}_{po}(S)\}$, $\Sigma = (S, F \cup P)$ and AX be the set of the following Horn clauses for P:

$$\begin{array}{ll} inv_s(c(x)) \Leftarrow inv_e(x), & c : e \to s \in C, \\ inv_e(\iota_i(x)) \Leftarrow inv_{e_i}(x), & e = \coprod_{i\in I} e_i \in \mathcal{T}_{po}(S),\ i \in I, \\ inv_e(x) \Leftarrow \bigwedge_{i\in I} inv_{e_i}(\pi_i(x)), & e = \prod_{i\in I} e_i \in \mathcal{T}_{po}(S), \\ inv_B(x), & B \in \mathcal{I}. \end{array}$$

Let $SP = (\Sigma, AX, \mathcal{C})$ and $\mathcal{A} \in Struct_{SP}$ be the algebra with carrier A such that for all $s \in S$,

$$inv_s^{\mathcal{A}} = \{g(x) \mid g \in \varphi_s^{\mathcal{C}}\}.$$

Suppose that $inv_s(x) \Rightarrow \varphi_s$ has been proved by fixpoint induction. Then

$$inv_s(x)^{lfp(\Phi_{SP})} \subseteq \varphi_s^{lfp(\Phi_{SP})}. \tag{1}$$

Hence

$$g(x) \in inv_s^{lfp(\Phi_{SP})} \Leftrightarrow g \in inv_s(x)^{lfp(\Phi_{SP})} \stackrel{(1)}{\Rightarrow} g \in \varphi_s^{lfp(\Phi_{SP})} = \varphi_s^{\mathcal{C}} \Leftrightarrow g(x) \in inv_s^{\mathcal{A}} \tag{2}$$

and thus

$$A \stackrel{Lemma\ 12.3\ (3)}{=} inv_s^{lfp(\Phi_{SP})} \stackrel{(2)}{\subseteq} inv_s^{\mathcal{A}} \subseteq A. \tag{3}$$

Therefore,

$$\varphi_s^{\mathcal{C}} = \{g \in A^V \mid g(x) \in inv_s^{\mathcal{A}}\} \stackrel{(3)}{=} \{g \in A^V \mid g(x) \in A\} = A^V,$$

i.e., $\mathcal{C}$ satisfies φ_s.

The number of generalizations of φ_s, i.e., applications of rule (2) in section 12.1 and consecutive extensions of axioms for the predicate q, in the proof of $inv_s(x) \Rightarrow \varphi_s$ by fixpoint induction agrees with the number of iterations of step 2 in the corresponding proof by algebraic induction (see section 12.2). Hence q is interpreted by the set B constructed in that proof.

12.5 Fixpoint induction upon a function

For verifying a recursive function f, i.e., showing properties of their input-output relation, we transform the Horn clauses for $f : e \to e'$ into Horn clauses for $p_f : e \times e' \to 2$—representing $graph(f)$ (see chapter 2)—by repeatedly applying the following transformation rules:

$$\frac{f(t) = u \ \Leftarrow \ \varphi}{p_f(t,u) \ \Leftarrow \ \varphi} \Updownarrow$$

$$\frac{\psi[f(v)/x] \ \Leftarrow \ \varphi}{\psi \ \Leftarrow \ p_f(v,x) \wedge \varphi} \Updownarrow \qquad \frac{\psi \ \Leftarrow \ \varphi[f(v)/x]}{\psi \ \Leftarrow \ \varphi \wedge p_f(v,x)} \Updownarrow$$

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:

$$(1) \qquad \frac{f(x) = y \Rightarrow \varphi}{\bigwedge_{p_f(t,u) \Leftarrow \delta \in rel(AX)} (\delta[q/p_f] \Rightarrow \varphi[t/x, u/y])} \Uparrow \quad f, p_f \not\in \varphi$$

After applying (1), the predicate q and the co-Horn clause $q(x,y) \Rightarrow \varphi$ are added to SP.

$$(2) \qquad \frac{q(x,y) \Rightarrow \varphi'}{\bigwedge_{p_f(t,u) \Leftarrow \delta \in rel(AX)} (\delta[q/p] \Rightarrow \varphi'[t/x, u/y])} \quad f, p_f, q \not\in \varphi'$$

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**:

$$\frac{p(u)(t) \;\Rightarrow\; \varphi}{p(z)(x) \;\Rightarrow\; (x = t \wedge z = u \Rightarrow \varphi)} \Updownarrow$$

$$\frac{f(t) = u \;\Rightarrow\; \varphi}{f(x) = y \;\Rightarrow\; (x = t \wedge y = u \Rightarrow \varphi)} \Updownarrow \qquad \frac{f(t) = u \wedge \varphi}{f(x) = y \;\Rightarrow\; (x = t \Rightarrow y = u \wedge \varphi)} \Updownarrow$$

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.

Sample proofs by fixpoint induction can be found in, e.g., [131, 138, 124].

12.6 Invariants are monotone

Suppose that for all $s \in S$, F contains a **constraint predicate** $\subset_s\ : \mathcal{P}(s)$. φ is **constraint compatible** if for all subformulas $\exists(x : e)\psi$ and $\forall(x : e)\psi$ of φ there is $\rho \in Fo_\Sigma(V)$ such that $\psi = (\subset_e(x) \wedge \rho)$ and $\psi = (\subset_e(x) \Rightarrow \rho)$, respectively.

Lemma 12.7

Let $\mathcal{A}$ be a Σ-algebra with carrier A, inv be a Σ-invariant of $\mathcal{A}$, $\mathcal{B} = \mathcal{A}|_{inv}$ (see section 9.9) 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) $\mathcal{A} \models \varphi$ implies $\mathcal{B} \models \varphi$.

Proof of (1) by induction on the size of φ.

Let $p : \mathcal{P}(e) \in P$ and $t : e \in \Lambda_\Sigma(V)$.

$$\begin{aligned} p(t)^{\mathcal{B}} &= \{g \in inv^V \mid t^{\mathcal{B}}(g) \in p^{\mathcal{B}}(g)\} = \{g \in A^V \mid g(V) \subseteq inv,\ t^{\mathcal{A}}(g) \in p^{\mathcal{A}}(g)\} \\ &= \{g \in p(t)^{\mathcal{A}} \mid g(V) \subseteq inv\}. \end{aligned}$$

Let $\varphi, \psi \in Fo_\Sigma(V)$, $s \in S$ and $x \in V_s$.

$$\begin{aligned} (\neg\varphi)^{\mathcal{B}} &= inv^V \setminus \varphi^{\mathcal{B}} \overset{ind.\ hyp.}{=} inv^V \setminus \{g \in \varphi^{\mathcal{A}} \mid g(V) \subseteq inv\} = inv^V \setminus \varphi^{\mathcal{A}} \\ &= \{g \in A^V \setminus \varphi^{\mathcal{A}} \mid g(V) \subseteq inv\} = \{g \in (\neg\varphi)^{\mathcal{A}} \mid g(V) \subseteq inv\}, \end{aligned}$$

$$\begin{aligned} (\varphi \wedge \psi)^{\mathcal{B}} &= \varphi^{\mathcal{B}} \cap \psi^{\mathcal{B}} \overset{ind.\ hyp.}{=} \{g \in \varphi^{\mathcal{A}} \mid g(V) \subseteq inv\} \cap \{g \in \psi^{\mathcal{A}} \mid g(V) \subseteq inv\} \\ &= \{g \in \varphi^{\mathcal{A}} \cap \psi^{\mathcal{A}} \mid g(V) \subseteq inv\} = \{g \in (\varphi \wedge \psi)^{\mathcal{A}} \mid g(V) \subseteq inv\}, \end{aligned}$$

$$
\begin{array}{ll}
 & (\forall(x:e)(\subset_e(x) \Rightarrow \varphi))^{\mathcal{B}} \\
= & \bigcap_{a\in inv_e}\{g \in inv^V \mid g[a/x] \in (\subset_e(x) \Rightarrow \varphi)^{\mathcal{B}}\} \\
\overset{ind.\ hyp.}{=} & \bigcap_{a\in inv_e}\{g \in inv^V \mid g[a/x](V) \subseteq inv,\ g[a/x] \in (\subset_e(x) \Rightarrow \varphi)^{\mathcal{A}}\} \\
= & \bigcap_{a\in inv_e}\{g \in inv^V \mid g[a/x](V) \subseteq inv,\ g[a/x] \notin \subset_e(x)^{\mathcal{A}} \vee g[a/x] \in \varphi^{\mathcal{A}}\} \\
= & \bigcap_{a\in inv_e}\{g \in inv^V \mid g[a/x](V) \subseteq inv,\ g[a/x](x) \notin \subset_e{}^{\mathcal{A}} \vee g[a/x] \in \varphi^{\mathcal{A}}\} \\
= & \bigcap_{a\in inv_e}\{g \in inv^V \mid g[a/x](V) \subseteq inv,\ a \notin inv \vee g[a/x] \in \varphi^{\mathcal{A}}\} \\
= & \bigcap_{a\in inv_e}\{g \in inv^V \mid g[a/x](V) \subseteq inv,\ g[a/x] \in \varphi^{\mathcal{A}}\} \\
= & \bigcap_{a\in inv_e}\{g \in inv^V \mid g[a/x] \in \varphi^{\mathcal{A}}\} \\
= & \{g \in inv^V \mid \forall\ a \in inv_e : g[a/x] \in \varphi^{\mathcal{A}}\} \\
= & \{g \in inv^V \mid \forall\ a \in A_e : (a \notin inv \vee g[a/x] \in \varphi^{\mathcal{A}})\} \\
= & \{g \in A^V \mid g(V) \subseteq inv,\ \forall\ a \in A_e : (a \notin inv \vee g[a/x] \in \varphi^{\mathcal{A}})\} \\
= & \bigcap_{a\in A_e}\{g \in A^V \mid g(V) \subseteq inv,\ a \notin inv \vee g[a/x] \in \varphi^{\mathcal{A}}\} \\
= & \bigcap_{a\in A_e}\{g \in A^V \mid g(V) \subseteq inv,\ g[a/x](x) \notin \subset_e{}^{\mathcal{A}} \vee g[a/x] \in \varphi^{\mathcal{A}}\} \\
= & \bigcap_{a\in A_e}\{g \in A^V \mid g(V) \subseteq inv,\ g[a/x] \notin \subset_e(x)^{\mathcal{A}} \vee g[a/x] \in \varphi^{\mathcal{A}}\}
\end{array}
$$

$$= \bigcap_{a \in A_e} \{g \in A^V \mid g(V) \subseteq inv,\ g[a/x] \in (\subset_e(x) \Rightarrow \varphi)^{\mathcal{A}}\}$$
$$= \{g \in (\forall(x : e)(\subset_e(x) \Rightarrow \varphi))^{\mathcal{A}} \mid g(V) \subseteq inv\}.$$

Proof of (2).

Suppose that $\mathcal{A}$ satisfies φ. Then $\varphi^{\mathcal{A}} = A^V$ and thus by (1),

$$\varphi^{\mathcal{B}} = \{g \in \varphi^{\mathcal{A}} \mid g(V) \subseteq inv\} = \{g \in inv^V \mid g \in \varphi^{\mathcal{A}}\} = \{g \in inv^V \mid g \in A^V\} = inv^V,$$

i.e., $\mathcal{B}$ satisfies φ. ❏

13 Coinduction rules

Let $\Sigma = (S, F \cup \{p : e' \to \mathcal{P}(e)\})$ be a signature, $\mathcal{C}$ be an (S,F)-algebra and $SP = (\Sigma, AX, \mathcal{C})$ be a co-Horn specification of P. For simplicity, we restrict ourselves to a single predicate p. The generalization to several predicates is straightforward (see Theorem 11.4 (2)).

13.1 Fixpoint coinduction upon a predicate

Let $p : e' \to \mathcal{P}(e) \in P$, φ be a closed (S,F)-λ-term, $x \in V_e$ and $z \in V_{e'}$.

A proof by fixpoint coinduction that $\mathcal{A} = gfp(\Phi_{SP})$ satisfies $\varphi \Rightarrow p(x)$ is a sequence $(\psi_1, \dots, \psi_n)$ of Σ-formulas such that the following conditions hold true:

- ψ_2 is the result of applying to ψ_1 the following rule:

$$(1) \qquad \frac{\varphi \Rightarrow p(z)(x)}{\bigwedge_{p(u)(t)\Rightarrow\delta\in AX}(\varphi[t/x, u/z] \Rightarrow \delta[q/p])} \Uparrow$$

  After applying (1), the predicate $q : e' \to \mathcal{P}(e)$ and the Horn clause $q(z)(x) \Leftarrow \varphi$ are added to SP.

- For all $1 < i < n$, ψ_{i+1} is the result of applying to ψ_i an expansion rule for $\mathcal{B}$ (see chapter 10) or the following rule:

$$(2) \qquad \frac{\varphi' \Rightarrow q(x)}{\bigwedge_{p(t)\Rightarrow\delta\in AX}(\varphi'[t/x] \Rightarrow \delta[q/p])}$$

  After applying (2), the Horn clause $q(z)(x) \Leftarrow \varphi'$ is added to SP.

- $\psi_n = \mathit{True}$.

(1) is an expansion rule for $\mathcal{A} = gfp(\Phi_{SP})$: If the succedent of (1) holds true in $\mathcal{A}$, then $\mathcal{A}$ satisfies the axioms for p if p were replaced by φ. Since $\mathcal{A}$ interprets p as the *greatest* relation satisfying the axioms for p, we conclude that the antecedent of (1) holds true in $\mathcal{A}$.

Proof sketch of the correctness of $(\psi_1, \dots, \psi_n)$

Suppose that he derivation $(\psi_1, \dots, \psi_n)$ contains k applications of (2). Then it reads schematically reads as follows:

$$
\begin{array}{ll}
& \varphi \Rightarrow p(z)(x) \\
\stackrel{(1)}{\vdash} & \bigwedge_{p(u)(t)\Rightarrow\delta\in AX}(\varphi[t/x] \Rightarrow \delta[q/p]) \qquad (*) \\
\stackrel{expansion\ rules}{\vdash} & \dots\ \varphi_1 \Rightarrow q(z)(x)\ \dots \\
\stackrel{(2)}{\vdash} & \dots\ \bigwedge_{p(u)(t)\Rightarrow\delta\in AX}(\varphi_1[t/x] \Rightarrow \delta[q/p])\ \dots \\
\vdash & \dots \\
\stackrel{expansion\ rules}{\vdash} & \dots\ \varphi_k \Rightarrow q(z)(x)\ \dots \\
\stackrel{(2)}{\vdash} & \dots\ \bigwedge_{p(u)(t)\Rightarrow\delta\in AX}(\varphi_k[t/x] \Rightarrow \delta[q/p])\ \dots \\
\stackrel{expansion\ rules}{\vdash} & True
\end{array}
$$

Since $q \notin \varphi \wedge \varphi_1 \wedge \dots \wedge \varphi_k$, $q(z)(x)$ is equivalent to φ 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, $q(z)(x)$ is considered to be equivalent to $\varphi \vee \varphi_1 \vee \dots \vee \varphi_k$. Then for all $1 \leq i \leq k$, $\varphi_i \Rightarrow q(z)(x)$ holds true, and thus the subderivation starting with $(*)$ yields the validity of

$$\bigwedge_{p(u)(t)\Rightarrow\delta\in AX} (\varphi[t/x,u/z] \Rightarrow \delta[q/p]) \tag{3}$$

and

$$\bigwedge_{p((u)t)\Rightarrow\delta\in AX} \bigwedge_{i=1}^{k} (\varphi_i[t/x,u/z] \Rightarrow \delta[q/p]). \tag{4}$$

(3)$\wedge$(4) is equivalent to

$$\bigwedge_{p(u)(t)\Rightarrow\delta\in AX} ((\varphi \vee \varphi_1 \vee \cdots \vee \varphi_k)[t/x,u/z] \Rightarrow \delta[q/p])$$

and thus to $\bigwedge_{p(u)(t)\Rightarrow\delta\in AX}(q[t/x,u/z] \Rightarrow \delta[q/p])$. Hence by q (instead of p) satisfies AX in $\mathcal{A}$ and thus by the correctness of (1),

$$q(z)(x) \Rightarrow p(z)(x) \tag{5}$$

and, in particular, the original goal $\varphi \Rightarrow p(z)(x)$ hold true in $\mathcal{A}$.

$q(x)$ can be regarded as a **generalization** of φ. By (5), $q(x)$ lies *somewhere* between φ and $p(x)$, the greatest (!) relation satisfying AX:

$$\varphi \;\Rightarrow\; q(x) \;\Rightarrow\; p(x).$$

Therefore, the validity of a coinductive conjecture like $\varphi \Rightarrow p(z)(x)$ 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 sections 11.5 and 13.6).

Let p be a *binary* predicate and $D\Sigma = (S, D)$ be a subsignature of Σ such that the set AX_p of co-Horn clauses for p consists of $D\Sigma$-bisimulation axioms, i.e., $\mathcal{A}$ satisfies AX_p iff $p^{\mathcal{B}}$ is a $D\Sigma$-bisimulation on $\mathcal{C}$. Then the antecedent of (1) reads as

$$\varphi \Rightarrow p(z)(x, y), \tag{6}$$

$p^{\mathcal{A}}$ is the *greatest* $D\Sigma$-bisimulation on $\mathcal{C}$, while the above-sketched derivation proves that $\mathcal{B} \in Struct_{\Sigma,\mathcal{C}}$ with $p^{\mathcal{B}} =_{def} (\varphi \vee \varphi_1 \vee \cdots \vee \varphi_k)^{\mathcal{B}}$ also satisfies AX_p, i.e., $p^{\mathcal{B}}$ is also a $D\Sigma$-bisimulation on $\mathcal{C}$. Hence $p^{\mathcal{B}} \subseteq p^{\mathcal{A}}$. If, in addition to the axioms for q that were added to AX after applications of (1) or (2), the Horn clauses

$$q(z)(x, x), \quad q(z)(x, y) \Leftarrow q(z)(y, x), \quad q(z)(x, y) \Leftarrow q(z)(x, x') \wedge q(z)(x', y) \tag{7}$$

were used in the proof of (6), q would actually denote the equivalence closure of $p^{\mathcal{B}}$, i.e.,

$$(p^{\mathcal{B}})^{eq} \subseteq p^{\mathcal{A}} \tag{8}$$

would actually be proved and not only $p^{\mathcal{B}} \subseteq p^{\mathcal{A}}$, which is also sufficient for (8):

Since $p^{\mathcal{B}} \subseteq p^{\mathcal{A}}$ implies $(p^{\mathcal{B}})^{eq} \subseteq (p^{\mathcal{A}})^{eq}$ and, by Theorem 9.6 (2), $p^{\mathcal{A}}$ is an equivalence relation and thus equal to $(p^{\mathcal{A}})^{eq}$, (8) indeed follows from $p^{\mathcal{B}} \subseteq p^{\mathcal{A}}$.

Let $D\Sigma$ be polynomial and destructive, $C\Sigma = (S', C')$ be a constructive signature of Σ, $\Sigma = C\Sigma \cup D\Sigma$, $C = \{c \in C \mid trg(c) \in S\}$ and $\mathcal{C}|_{D\Sigma}$ be final in $Alg_{D\Sigma}$. such that the assumptions of Theorem 16.3 hold true. Then ****

$$\bigwedge_{c:e_c \to s \in C,\ d:s \to e \in D} \forall x_{c,1} \dots \forall x_{c,n_c}\ d(c(x_c)) = t_{c,d} : e$$

is a biinductive definition of C. If, in addition to the axioms for q that were added to AX after applications of (1) or (2), (7) and

$$q(c(x), c(y)) \Leftarrow q(x, y), \qquad c \in C, \tag{9}$$

were used in the proof of (6), then q would actually denote the $C\Sigma$-congruence closure of $p^{\mathcal{B}}$, i.e.,

$$p^{\mathcal{B}}_C \subseteq p^{\mathcal{A}} \tag{10}$$

would actually be proved and not only $p^{\mathcal{B}} \subseteq p^{\mathcal{A}}$, which is also sufficient for (10):

Since $p^{\mathcal{B}}$ is a $D\Sigma$-bisimulation on $\mathcal{C}$ and thus a $D\Sigma$-bisimulation modulo C, Lemma 16.4 implies that $p^{\mathcal{B}}_C$ is a $D\Sigma$-congruence and thus a $D\Sigma$-bisimulation. Since $p^{\mathcal{A}}$ is the *greatest* one, $p^{\mathcal{B}}_C \subseteq p^{\mathcal{A}}$. Hence (10) indeed follows from $p^{\mathcal{B}} \subseteq p^{\mathcal{A}}$.

If (9) is used in a proof of $\varphi \Rightarrow p(z)(x)$ by fixpoint coinduction, the proof is called a proof by (**fixpoint**) **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 13.5).

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**:

$$\frac{\varphi \;\Rightarrow\; p(u)(t)}{\varphi \wedge x = t \wedge z = u \;\Rightarrow\; p(z)(x)} \;\Updownarrow$$

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.

Sample proofs by fixpoint coinduction can be found in, e.g., [131, 138].

Coinductive logic programming or *co-logic programming* [63, 166] 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 φ, but also compares φ with all predecessors of φ 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.

13.2 Congruences and algebraic coinduction

Let $\Sigma = (S, F)$ be a signature, $\mathcal{A}$ be a Σ-algebra with carrier A and R be a Σ-congruence on $\mathcal{A}$.

$nat_R : \mathcal{A} \to \mathcal{A}/R$ denotes the Σ-homomorphic natural map (see section 9.1).

Lemma 13.1 (Homomorphisms and congruences)

(1) Let $h : A \to B$ be an S-sorted function and $\mathcal{A}$ be a Σ-algebra with carrier A. B can be extended to a Σ-algebra $\mathcal{B}$ and h to a Σ-homomorphism from $\mathcal{A}$ to $\mathcal{B}$ iff $ker(h)$ is a Σ-congruence.

(2) $h : \mathcal{A} \to \mathcal{B}$ is Σ-homomorphic iff there is a unique Σ-monomorphism $h' : \mathcal{A}/ker(h) \to \mathcal{B}$ with $h' \circ nat_{ker(h)} = h$.

Hence, if h is epi, then by Lemma 4.1 (1), h' is epi and thus $\mathcal{A}/ker(h)$ and $\mathcal{B}$ are Σ-isomorphic.

Proof.

(1) If h is Σ-homomorphic, then $ker(h)$ is a Σ-congruence. Let $ker(h)$ be a Σ-congruence. For all $f : e \to e' \in F$, define $f^{\mathcal{B}} : B_e \to B_{e'}$ 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 Σ-algebra and h is Σ-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 9.1 (1). ❑

Moreover, by Theorem 3.4 (2), for every S-sorted binary relation R on A, the greatest Σ-congruence contained in R is the intersection of all R_n, $n \in \mathbb{N}$, with $R_0 = R$ and for all $s \in S$,

$$R_{n+1,s} = \{(a,b) \in A^2 \mid \forall\, d : s \to e \in D : (d^{\mathcal{A}}(a), d^{\mathcal{A}}(b)) \in R^{eq}_{n,e}\}.$$

Proofs by algebraic coinduction

Let $D \subseteq F$ be a set of destructors and $D\Sigma = (S, D)$.

- A Σ-algebra $\mathcal{A}$ with carrier A **satisfies the (algebraic) coinduction principle for $D\Sigma$** 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 Σ-algebra $\mathcal{A}$ with carrier A that satisfies the coinduction principle for $D\Sigma$ and for all $s \in S$, $E_s \subseteq \Lambda_\Sigma(V)^2_s$.

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 $R_s := R_{0,s} =_{def} \{(t^{\mathcal{A}}(g), u^{\mathcal{A}}(g)) \mid (t,u) \in E_s,\ g \in A^V\}$.

- **Step 2**: For all $s \in S$, let

$$R'_s = \{(a,b) \in A_s^2 \mid \forall\, d : s \to e \in D : (d^{\mathcal{A}}(a), d^{\mathcal{A}}(b)) \in R_e^{eq}\}.$$

- **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 $D\Sigma$, $R_0 \subseteq R^{eq} \subseteq \Delta_A$. Hence for all $s \in S$,

$$\{(t^{\mathcal{A}}(g), u^{\mathcal{A}}(g)) \mid (t,u) \in E_s,\ g \in A^V\} = R_{0,s} \subseteq \Delta_{A_s},$$

  i.e., $\mathcal{A}$ satisfies E.
  If $R \not\subseteq R'$, then for all $e \in \mathcal{T}_{po}(S)$, set

$$R_e := R_e \cup \{(d^{\mathcal{A}}(a), d^{\mathcal{A}}(b)) \mid (a,b) \in R_s,\ d : s \to e \in D\}$$

  and go to Step 2.

Lemma 13.2

Let $h : \mathcal{A} \to \mathcal{B}$ be Σ-homomorphic and R be a Σ-congruence on $\mathcal{A}$.

$$h(R) \;=\; \{(h(a), h(b)) \mid (a,b) \in R\}$$

is a Σ-congruence on $\mathcal{B}$.

Proof. Let $f : e \to e' \in F$ 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 Σ-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 $(f^{\mathcal{B}}(c), f^{\mathcal{B}}(d)) \in h(R)_{e'}$. ❏

Lemma 13.3 (Coinduction and finality)

Let $\Sigma = (S, F)$ be a destructive signature and $\mathcal{A}, \mathcal{B}$ be Σ-algebras with carriers A, B and $\mathcal{K}$ be a full subcategory $\mathcal{K}$ of Alg_Σ that is closed under quotients.

(1) $\mathcal{A}$ satisfies the coinduction principle iff Δ_A is the only Σ-congruence on $\mathcal{A}$.

(2) If Δ_A is the only Σ-congruence on $\mathcal{A}$, then all Σ-homomorphisms from $\mathcal{B}$ to $\mathcal{A}$ coincide.

(3) $\mathcal{A}$ is final in $\mathcal{K}$ iff Δ_A is the only Σ-congruence on $\mathcal{A}$ and for all $\mathcal{B} \in \mathcal{K}$ there is a Σ-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 Σ-homomorphism $unfold^{\mathcal{B}} : \mathcal{B} \to \mathcal{A}$ is the greatest Σ-congruence on $\mathcal{B}$ ([157], Prop. 2.7).

Proof.

(1) "⇒": Suppose that $\mathcal{A}$ satisfies the coinduction principle and R is a Σ-congruence on $\mathcal{A}$. Hence $R \subseteq \Delta_A$. Since R is reflexive and thus $\Delta_A \subseteq R$, R agrees with Δ_A .

"⇐": Suppose that R is a Σ-congruence that contains a binary relation R' on A and Δ_A is the only Σ-congruence on $\mathcal{A}$. Then $R' \subseteq R = \Delta_A$.

(2) Let $g, h : \mathcal{B} \to \mathcal{A}$ be Σ-homomorphisms. Then $R = \{(g(b), h(b)) \mid b \in B\}$ is a Σ-congruence on A: Let $f : e \to e' \in F$, $b \in B_e$ and $(g_e(b), h_e(b)) \in R_e$. Since g and h are Σ-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 7.2 (4) implies

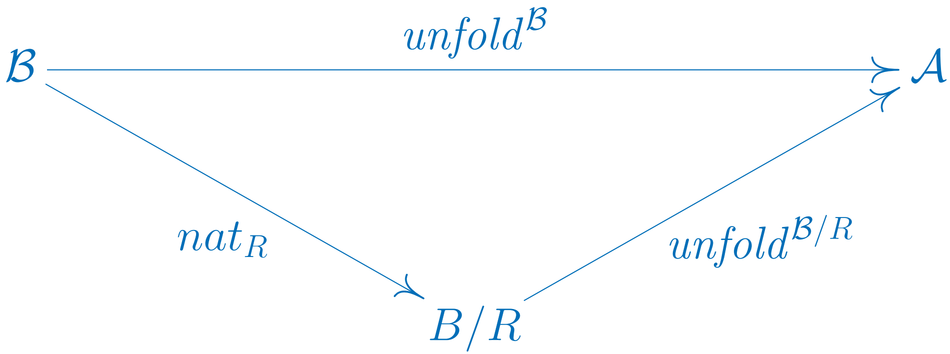
$$(f^{\mathcal{A}}(g_e(b)), f^{\mathcal{A}}(h_e(b))) = (g_{e'}(f^{\mathcal{B}}(b)), h_{e'}(f^{\mathcal{B}}(b))) \in R_{e'}.$$

Since Δ_A is the only Σ-congruence on $\mathcal{A}$, R agrees with Δ_A and thus for all $b \in B$, $g(b) = h(b)$.

(3) "⇒": Let $\mathcal{A}$ be final in $\mathcal{K}$ and R be a Σ-congruence on $\mathcal{A}$. R induces the Σ-epimorphism $nat_R : \mathcal{A} \to \mathcal{A}/R$. Hence Lemma 4.3 (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 Σ-homomorphism from $\mathcal{B}$ to $\mathcal{A}$.

"⇐": Suppose that Δ_A is the only Σ-congruence on $\mathcal{A}$ and for all $\mathcal{B} \in \mathcal{K}$ there is a Σ-homomorphism $h : \mathcal{B} \to \mathcal{A}$. By (2), h is unique. Hence $\mathcal{A}$ is final in $\mathcal{K}$.

(4) Let R be a Σ-congruence on $\mathcal{B}$. Since $\mathcal{A}$ is final in $\mathcal{K}$, the following diagram commutes:



Hence for all $b, c \in B$,

$$(b,c) \in R \;\Rightarrow\; [b]_R = [c]_R \;\Rightarrow\; unfold^{\mathcal{B}}(b) = unfold^{\mathcal{B}/R}([b]_R) = unfold^{\mathcal{B}/R}([c]_R)$$
$$= unfold^{\mathcal{B}}(c).$$

We conclude that $ker(unfold^{\mathcal{B}})$ contains R.

Alternative proof of (4): By Lemma 13.2,

$$unfold^{\mathcal{B}}(R) \;=\; \{(unfold^{\mathcal{B}}(b), unfold^{\mathcal{B}}(c)) \mid (b,c) \in R\}$$

is a Σ-congruence on $\mathcal{A}$. By (3), Δ_A is the only Σ-congruence on $\mathcal{A}$. Hence $unfold^{\mathcal{B}}(R) = \Delta_A$ and thus for all $(b,c) \in R$, $unfold^{\mathcal{B}}(b) = unfold^{\mathcal{B}}(c)$, i.e., $(b,c) \in ker(unfold^{\mathcal{B}})$. ❑

13.3 Algebraic coinduction as fixpoint coinduction

Let $D \subseteq F$ be a set of destructors, $D\Sigma = (S, D)$, $\mathcal{C}$ be a final $D\Sigma$-algebra and for all $s \in S$, $E_s \subseteq \Lambda_\Sigma(V)^2_s$.

Moreover, let $P = \{\sim_e : e \times e \mid e \in \mathcal{T}_{po}(S)\}$, $\Sigma' = (S, F \cup P)$ and AX be the following set of co-Horn clauses for P:

$$
\begin{array}{ll}
x \sim_s y \Rightarrow d(x) \sim_e d(y), & d : s \to e \in D, \\
\iota_i(x) \sim_e \iota_i(y) \Rightarrow x \sim_{e_i} y, & e = \coprod_{i \in I} e_i \in \mathcal{T}_{po}(S),\ i \in I, \\
\neg(\iota_i(x) \sim_e \iota_j(y)), & e = \coprod_{i \in I} e_i \in \mathcal{T}_{po}(S),\ i, j \in I,\ i \neq j, \\
x \sim_e y \Rightarrow \pi_i(x) \sim_{e_i} \pi_i(y), & e = \prod_{i \in I} e_i \in \mathcal{T}_{po}(S),\ i \in I, \\
x \sim_B y \Rightarrow x = y, & B \in \mathcal{I}.
\end{array}
$$

Let $SP = (\Sigma, AX, \mathcal{C})$ and $\mathcal{A} \in Struct_{SP}$ be the algebra with carrier A such that for all $s \in S$,

$$\sim_s^{\mathcal{A}} \;=\; \{(t^{\mathcal{C}}(g), u^{\mathcal{C}}(g)) \mid t = u \in E_s,\ g \in A^V\}$$

Let $x, y \in V_s \setminus free(E_s)$ and

$$\varphi_s \;=\; \bigvee_{t=u \in E_s} \exists\, free(E_s) : (x = t \wedge y = u).$$

Suppose that $\varphi_s \Rightarrow x \sim_s y$ has been proved by fixpoint coinduction. Then

$$\varphi_s^{gfp(\Phi_{SP})} \subseteq (x \sim_s y)^{gfp(\Phi_{SP})}. \tag{1}$$

Hence

$$\begin{aligned} & g(x) \sim_s^{\mathcal{A}} g(y) \Leftrightarrow \bigvee_{t=u \in E_s} (g(x) = t^{\mathcal{C}}(g) \wedge g(y) = u^{\mathcal{C}}(g)) \Leftrightarrow g \in \varphi_s^{gfp(\Phi_{SP})} \\ & \overset{(1)}{\Rightarrow} g \in (x \sim_s y)^{gfp(\Phi_{SP})} \Leftrightarrow g(x) \sim_s^{gfp(\Phi_{SP})} g(y) \end{aligned} \tag{2}$$

and thus

$$\sim_s^{\mathcal{A}} \overset{(2)}{\subseteq} \sim_s^{gfp(\Phi_{SP})} \overset{Lemma\ 13.3\ (3)}{=} \Delta_A. \tag{3}$$

Let $t = u \in E_s$. Then for all $g \in A^V$, $t^{\mathcal{C}}(g) \sim_s^{\mathcal{A}} u^{\mathcal{C}}(g)$ and thus by (3), $t^{\mathcal{C}}(g) = u^{\mathcal{C}}(g)$. Therefore,

$$(t = u)^{\mathcal{C}} = \{g \in A^V \mid t^{\mathcal{C}}(g) = u^{\mathcal{C}}(g)\} = A^V,$$

i.e., $\mathcal{C}$ satisfies E_s.

The number of generalizations of φ_s, i.e., applications of rule (2) in section 13.1and consecutive extensions of axioms for the predicate q, in the proof of $\varphi_s \Rightarrow x \sim_s y$ by fixpoint coinduction agrees with the number of iterations of step 2 in the corresponding proof by algebraic coinduction (see section 13.2). Hence q is interpreted by the relation R constructed in that proof.

Example 13.4 Let $\Sigma = \mathit{Stream}(\mathbb{Z}) \cup F$ (see section 8.3) where

$$
\begin{aligned}
F &= \{\mathit{zeros}, \mathit{ones}, \mathit{blink} : 1 \to \mathit{list}\} \cup \\
&\quad \{\mathit{cons} : \mathbb{N} \times \mathit{list} \to \mathit{list},\ \mathit{zip} : \mathit{list} \times \mathit{list} \to \mathit{list},\ \mathit{evens} : \mathit{list} \to \mathit{list}\}.
\end{aligned}
$$

Let $\mathcal{A}$ be a Σ-algebra with carrier A such that $\mathcal{A}|_{\mathit{Stream}(\mathbb{Z})}$ is final in $\mathit{Alg}_{\mathit{Stream}(\mathbb{Z})}$ and satisfies the following equations:

$$
\begin{aligned}
&\mathit{head}(\mathit{zeros}) = 0, && \mathit{tail}(\mathit{zeros}) = \mathit{zeros}, \\
&\mathit{head}(\mathit{ones}) = 1, && \mathit{tail}(\mathit{ones}) = \mathit{ones}, \\
&\mathit{head}(\mathit{blink}) = 0, && \mathit{tail}(\mathit{blink}) = \mathit{cons}(1, \mathit{blink}), \\
&\mathit{head}(\mathit{cons}(x, s)) = x, && \mathit{tail}(\mathit{cons}(x, s)) = s, \\
&\mathit{head}(\mathit{zip}(s, s')) = \mathit{head}(s'), && \mathit{tail}(\mathit{zip}(s, s')) = \mathit{zip}(s', s), \\
&\mathit{head}(\mathit{evens}(s)) = \mathit{head}(s), && \mathit{tail}(\mathit{evens}(s)) = \mathit{tail}(\mathit{tail}(s)).
\end{aligned}
$$

We show by algebraic coinduction that $\mathcal{A}$ satisfies the equations

$$
\begin{aligned}
\mathit{zip}(\mathit{zeros}, \mathit{ones}) &= \mathit{blink}, && (1) \\
\mathit{evens}(\mathit{zip}(s, s')) &= s. && (2)
\end{aligned}
$$

Proof of (1). Let

$$a = zip(zeros, ones)^{\mathcal{A}},\ \ b = blink^{\mathcal{A}},$$

$$R = \{(a, b)\},$$

$$R' = \{(a, b) \in A^2 \mid head^{\mathcal{A}}(a) = head^{\mathcal{A}}(b),\ (tail^{\mathcal{A}}(a), tail^{\mathcal{A}}(b)) \in R^{eq}\}.$$

We have

$$head^{\mathcal{A}}(a) = head^{\mathcal{A}}(zip^{\mathcal{A}}(zeros^{\mathcal{A}}, ones^{\mathcal{A}})) = head^{\mathcal{A}}(blink^{\mathcal{A}}) = head^{\mathcal{A}}(b), \quad (3)$$

$$tail^{\mathcal{A}}(a) = tail^{\mathcal{A}}(zip^{\mathcal{A}}(zeros^{\mathcal{A}}, ones^{\mathcal{A}})) = zip^{\mathcal{A}}(ones^{\mathcal{A}}, zeros^{\mathcal{A}}), \quad (4)$$

$$tail^{\mathcal{A}}(b) = tail^{\mathcal{A}}(blink^{\mathcal{A}}) = cons^{\mathcal{A}}(1, blink^{\mathcal{A}}). \quad (5)$$

By (4) and (5), $(tail^{\mathcal{A}}(a), tail^{\mathcal{A}}(b)) \notin R^{eq}$ and thus $(a, b) \notin R'$. Hence we extend R in accordance with Step 3 of the coinductive-proof procedure:

$$a' = zip^{\mathcal{A}}(ones^{\mathcal{A}}, zeros^{\mathcal{A}}),\ \ b' = cons^{\mathcal{A}}(1, blink^{\mathcal{A}}),$$

$$R = \{(a, b), (a', b')\},$$

$$R' = \{(a, b) \in A^2 \mid head^{\mathcal{A}}(a) = head^{\mathcal{A}}(b),\ (tail^{\mathcal{A}}(a), tail^{\mathcal{A}}(b)) \in R^{eq}\}.$$

By (3), (4) and (5), $(a, b) \in R'$. Moreover,

$$head^{\mathcal{A}}(a') = head^{\mathcal{A}}(zip^{\mathcal{A}}(ones^{\mathcal{A}}, zeros^{\mathcal{A}}))) = 1 = head^{\mathcal{A}}(cons^{\mathcal{A}}(1, blink^{\mathcal{A}}))$$

$$= head^{\mathcal{A}}(b'),$$

$$tail^{\mathcal{A}}(a') = tail^{\mathcal{A}}(zip^{\mathcal{A}}(ones^{\mathcal{A}}, zeros^{\mathcal{A}})) = zip^{\mathcal{A}}(zeros^{\mathcal{A}}, tail^{\mathcal{A}}(ones^{\mathcal{A}}))$$
$$= zip^{\mathcal{A}}(zeros^{\mathcal{A}}, ones^{\mathcal{A}}),$$
$$tail^{\mathcal{A}}(b') = tail^{\mathcal{A}}(cons^{\mathcal{A}}(1, blink^{\mathcal{A}})) = blink^{\mathcal{A}}.$$

Hence $(tail^{\mathcal{A}}(a'), tail^{\mathcal{A}}(b')) \in R$ and thus $(a', b') \in R'$. Consequently, $R \subseteq R'$ and thus by the coinduction principle for $\mathit{Stream}(\mathbb{Z})$, $R \subseteq \Delta_A$, i.e., $\mathcal{A}$ satisfies (1).

Proof of (2). Let

$$f = evens(zip(s, s'))^{\mathcal{A}},$$
$$R = \{(f(g), g(s)) \mid g \in A^V\},$$
$$R' = \{(a, b) \in A^2 \mid head^{\mathcal{A}}(a) = head^{\mathcal{A}}(b),\ (tail^{\mathcal{A}}(a), tail^{\mathcal{A}}(b)) \in R^{eq}\}.$$

We have

$$head^{\mathcal{A}}(f(g)) = head^{\mathcal{A}}(evens^{\mathcal{A}}(zip^{\mathcal{A}}(g(s), g(s')))) = head^{\mathcal{A}}(zip^{\mathcal{A}}(g(s), g(s')))$$
$$= head^{\mathcal{A}}(g(s)), \tag{6}$$
$$tail^{\mathcal{A}}(f(g)) = tail^{\mathcal{A}}(evens^{\mathcal{A}}(zip^{\mathcal{A}}(g(s), g(s'))))$$
$$= evens^{\mathcal{A}}(tail^{\mathcal{A}}(tail^{\mathcal{A}}(zip^{\mathcal{A}}(g(s), g(s'))))) = evens^{\mathcal{A}}(tail^{\mathcal{A}}(zip^{\mathcal{A}}(g(s'), tail(g(s)))))$$
$$= evens^{\mathcal{A}}(zip^{\mathcal{A}}(tail^{\mathcal{A}}(g(s)), tail^{\mathcal{A}}(g(s')))) = f(tail^{\mathcal{A}} \circ g). \tag{7}$$

Hence by (7), $(tail^{\mathcal{A}}(f(g)), tail^{\mathcal{A}}(g(s))) \in R$ and thus by (6), $(f(g), g(s)) \in R'$. Consequently, $R \subseteq R'$ and thus by the coinduction principle for $Stream(\mathbb{Z})$, $R \subseteq \Delta_A$, i.e., $\mathcal{A}$ satisfies (2).

Proofs of (1) and (2) by fixpoint coinduction, performed by Expander2 with respect to the specification stream, can be found here. ❏

13.4 Coinduction modulo constructors

Let $D \subseteq F$ be a set of destructors, $D\Sigma = (S, D)$ and $C \subseteq F$ be a set of constructors. Under the assumptions of Lemma 16.4, the coinduction principle for $D\Sigma$ (see above) can be weakened as follows:

❀ A Σ-algebra A with carrier A **satisfies the coinduction principle for $D\Sigma$ 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 coinductive-proof procedure of section 13.2 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
$$R'_s = \{(a,b) \in A_s^2 \mid \forall\, d : s \to e \in D : (d^{\mathcal{A}}(a), d^{\mathcal{A}}(b)) \in R_{C,e}\}.$$

If $R \subseteq R'$, then R is a $D\Sigma$-bisimulation modulo C. Hence by Lemma 16.4, the $C\Sigma$-congruence closure R_C of R 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 $s \in S$,
$$\{(t^{\mathcal{A}}(g), u^{\mathcal{A}}(g)) \mid (t,u) \in E_s,\ g \in A^V\} = R_{0,s} \subseteq \Delta_{A_s},$$
i.e., $\mathcal{A}$ satisfies E.

Example 13.5

Let $\Sigma = Acc(X) \cup Reg(X)$ and $\mathcal{A}$ be the Σ-algebra with carrier A, $\mathcal{A}|_{Reg(X)} = Pow(X)$ (sample algebra 9.6.20) and $\mathcal{A}|_{Acc(X)} = Lang(X)$ (sample algebra 9.6.19).

The biinductive definition of the Brzozowski automaton (see sample biinductive definition 16.5.6) provide the assumptions of Lemma 16.4. We show that $\mathcal{A}$ satisfies the distributive law
$$x * (y+z) \;=\; (x*y)+(x*z). \qquad (1)$$

The following proof by algebraic coinduction modulo $C = \{par\}$ uses the equations that define the Brzozowski automaton (see sample algebra 9.6.23) and the equations

$$
\begin{aligned}
\widehat{0} * x &= \widehat{0}, && (2)\\
x * \widehat{1} &= x, && (3)\\
\widehat{1} * x &= x, && (4)\\
(x_1 + y_1) + (x_2 + y_2) &= (x_1 + x_2) + (y_1 + y_2). && (5)
\end{aligned}
$$

Let $x' \in X$,

$$
\begin{aligned}
& f = (x * (y + z))^{\mathcal{A}}, \qquad f' = ((x * y) + (x * z))^{\mathcal{A}},\\
& a = \delta^{\mathcal{A}}(g(x))(x') *^{\mathcal{A}} (g(y) +^{\mathcal{A}} g(z)),\\
& b = \delta^{\mathcal{A}}(g(x))(x') *^{\mathcal{A}} g(y), \qquad c = \delta^{\mathcal{A}}(g(x))(x') *^{\mathcal{A}} g(z),\\
& d = \delta^{\mathcal{A}}(g(y))(x') +^{\mathcal{A}} \delta^{\mathcal{A}}(g(z))(x'),\\
& R = \{(f(g), f'(g)) \mid g \in A^V\},\\
& R' = \{(a, b) \in A^2 \mid \beta^{\mathcal{A}}(a) = \beta^{\mathcal{A}}(b),\ (\delta^{\mathcal{A}}(a), \delta^{\mathcal{A}}(b)) \in R_C\}.
\end{aligned}
$$

Then

$$
(a, b +^{\mathcal{A}} c) = (f(g[\delta^{\mathcal{A}}(g(x))(x')/x]), f'(g[\delta^{\mathcal{A}}(g(x))(x')/x])) \in R. \tag{6}
$$

Moreover,

$$
\begin{aligned}
\beta^{\mathcal{A}}(f(g)) &= \beta^{\mathcal{A}}(g(x) *^{\mathcal{A}} (g(y) +^{\mathcal{A}} g(z))) = \beta^{\mathcal{A}}(g(x)) * \beta^{\mathcal{A}}(g(y) +^{\mathcal{A}} \beta^{\mathcal{A}}(g(z))) \\
&= \beta^{\mathcal{A}}(g(x)) * max(\beta^{\mathcal{A}}(g(y)), \beta^{\mathcal{A}}(g(z))) \\
&= max(\beta^{\mathcal{A}}(g(x)) * \beta^{\mathcal{A}}(g(y)), \beta^{\mathcal{A}}(g(x)) * \beta^{\mathcal{A}}(g(z))) \\
&= max(\beta^{\mathcal{A}}(g(x) *^{\mathcal{A}} g(y)), \beta^{\mathcal{A}}(g(x)^{\mathcal{A}} g(z))) \\
&= \beta^{\mathcal{A}}((g(x) *^{\mathcal{A}} g(y)) +^{\mathcal{A}} (g(x) *^{\mathcal{A}} g(z))) = \beta^{\mathcal{A}}(f'(g)).
\end{aligned}
\tag{7}
$$

$$
\begin{aligned}
\delta^{\mathcal{A}}(f(g))(x') &= \delta^{\mathcal{A}}(g(x) *^{\mathcal{A}} (g(y) +^{\mathcal{A}} g(z)))(x') \\
&= a +^{\mathcal{A}} (\widehat{\beta^{\mathcal{A}}(g(x))}^{\mathcal{A}} *^{\mathcal{A}} (\delta^{\mathcal{A}}(g(y))(x') +^{\mathcal{A}} \delta^{\mathcal{A}}(g(z))(x'))) \\
&\overset{(2/3/4)}{=} \begin{cases} a & \text{if } \beta^{\mathcal{A}}(g(x)) = 0 \\ a +^{\mathcal{A}} d & \text{otherwise} \end{cases}
\end{aligned}
\tag{8}
$$

$$\begin{aligned}
\delta^{\mathcal{A}}(f'(g))(x') &= \delta^{\mathcal{A}}((g(x) *^{\mathcal{A}} g(y)) +^{\mathcal{A}} (g(x) *^{\mathcal{A}} g(z)))(x') \\
&= \delta^{\mathcal{A}}(g(x) *^{\mathcal{A}} g(y))(x') +^{\mathcal{A}} \delta^{\mathcal{A}}(g(x) *^{\mathcal{A}} g(z))(x') \\
&= (b +^{\mathcal{A}} (\widehat{\beta^{\mathcal{A}}(g(x))}^{\mathcal{A}} *^{\mathcal{A}} \delta^{\mathcal{A}}(g(y))(x'))) +^{\mathcal{A}} (c +^{\mathcal{A}} (\widehat{\beta^{\mathcal{A}}(g(x))}^{\mathcal{A}} *^{\mathcal{A}} \delta^{\mathcal{A}}(g(z))(x'))) \\
&\stackrel{(2/3/4)}{=} \begin{cases} b +^{\mathcal{A}} c & \text{if } \beta^{\mathcal{A}}(g(x)) = 0 \\ (b +^{\mathcal{A}} \delta^{\mathcal{A}}(g(y))(x')) +^{\mathcal{A}} (c +^{\mathcal{A}} \delta^{\mathcal{A}}(g(z))(x')) & \text{otherwise} \end{cases} \\
&\stackrel{(5)}{=} \begin{cases} b +^{\mathcal{A}} c & \text{if } \beta^{\mathcal{A}}(g(x)) = 0 \\ (b +^{\mathcal{A}} c) +^{\mathcal{A}} d & \text{otherwise} \end{cases} \qquad (9)
\end{aligned}$$

By (8) and (9),

$$(\delta^{\mathcal{A}}(f(g))(x'), \delta^{\mathcal{A}}(f'(g))(x')) = \begin{cases} (a, b +^{\mathcal{A}} c) & \text{if } \beta^{\mathcal{A}}(g(x)) = 0 \\ (a +^{\mathcal{A}} d, (b +^{\mathcal{A}} c) +^{\mathcal{A}} d & \text{otherwise.} \end{cases}$$

Hence $(\delta^{\mathcal{A}}(f(g))(x'), \delta^{\mathcal{A}}(f'(g))(x')) \in R_C$ and thus by (7), $(f(g), f'(g)) \in R'$. Consequently, $R \subseteq R'$ and thus by the coinduction principle for $Acc(X)$ modulo C, $R \subseteq \Delta_A$, i.e., $\mathcal{A}$ satisfies (1).

A proof of (1) by fixpoint coinduction modulo C, performed by Expander2 with respect to the specification brozowski, can be found here. ❑

13.5 Quotients are monotone

Lemma 13.6

Let $\mathcal{A}$ be a Σ-algebra with carrier A, R be a symmetric Σ-congruence on $\mathcal{A}$, $\mathcal{B} = \mathcal{A}/R$ (see section 9.10), $\varphi \in Fo_\Sigma(V)$ and $f, g \in A^V$ such that for all $x \in V$, $(f(x), g(x)) \in R$.

(1) $f \in \varphi^\mathcal{A} \Leftrightarrow g \in \varphi^\mathcal{A}$.

(2) $\varphi^\mathcal{B} = \{nat_R \circ g \mid g \in \varphi^\mathcal{A}\}$.

(3) $\mathcal{A} \models \varphi$ implies $\mathcal{B} \models \varphi$.

Proof of (1) by induction on the size of φ.

Let $e \in \mathcal{T}_{fo}(S)$, $p : \mathcal{P}(e) \in P$, $t : e \in \Lambda_\Sigma(V)$ and $f \in p(t)^\mathcal{A}$. Then $t^\mathcal{A}(f) \in p^\mathcal{A}$. By assumption and straightforward induction on the size of t, $(t^\mathcal{A}(f), t^\mathcal{A}(g)) \in R_e$. Hence $t^\mathcal{A}(g) \in p^\mathcal{A}$ and thus $g \in p(t)^\mathcal{A}$.

Let $\varphi, \psi \in Fo_\Sigma(V)$, $e \in \mathcal{T}_{fo}(S)$ and $x \in V_e$.

$$f \in (\neg\varphi)^\mathcal{A} \Leftrightarrow f \in A^V \setminus \varphi^\mathcal{A} \overset{ind.\ hyp.}{\Leftrightarrow} g \in A_e \setminus \varphi^\mathcal{A} \Leftrightarrow g \in (\neg\varphi)^\mathcal{A},$$

$$f \in (\varphi \wedge \psi)^\mathcal{A} \Leftrightarrow f \in \varphi^\mathcal{A} \wedge f \in \psi^\mathcal{A} \overset{ind.\ hyp.}{\Leftrightarrow} g \in \varphi^\mathcal{A} \wedge g \in \psi^\mathcal{A} \Leftrightarrow g \in (\varphi \wedge \psi)^\mathcal{A},$$

$$f \in (\forall x\varphi)^\mathcal{A} \Leftrightarrow \forall\ a \in A_e : f[a/x] \in \varphi^\mathcal{A} \overset{ind.\ hyp.}{\Leftrightarrow} \forall\ a \in A_e : g[c/x] \in \varphi^\mathcal{A} \Leftrightarrow g \in (\forall x\varphi)^\mathcal{A}.$$

Proof of (2) by induction on the size of φ.

Let $e \in \mathcal{T}_{fo}(S)$, $p : \mathcal{P}(e) \in P$, $t : e \in \Lambda_\Sigma(V)$.

$$\begin{aligned} p(t)^{\mathcal{B}} &= \{f \in (A/R)^V \mid t^{\mathcal{A}}(f) \in p^{\mathcal{B}}\} = \{nat_R \circ g \mid g \in A^V,\ t^{\mathcal{B}}(nat_R \circ g) \in p^{\mathcal{B}}\} \\ &= \{nat_R \circ g \mid g \in A^V,\ nat_R(t^{\mathcal{A}}(g)) \in p^{\mathcal{B}}\} = \{nat_R \circ g \mid g \in A^V,\ [t^{\mathcal{A}}(g)]_R \in p^{\mathcal{B}}\} \\ &= \{nat_R \circ g \mid g \in A^V,\ t^{\mathcal{A}}(g) \in p^{\mathcal{A}}\} = \{nat_R \circ g \mid g \in p(t)^{\mathcal{A}}\}. \end{aligned}$$

Let $\varphi, \psi \in Fo_\Sigma(V)$, $e \in \mathcal{T}_{fo}(S)$ and $x \in V_e$.

$$\begin{aligned} (\neg\varphi)^{\mathcal{B}} &= (A/R)^V \setminus \varphi^{\mathcal{B}} \overset{ind.\ hyp.}{=} (A/R)^V \setminus \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}}\} \\ &= \{nat_R \circ g \mid g \in A^V \setminus \varphi^{\mathcal{A}}\} = \{nat_R \circ g \mid g \in (\neg\varphi)^{\mathcal{A}}\}, \\ (\varphi \wedge \psi)^{\mathcal{B}} &= \varphi^{\mathcal{B}} \cap \psi^{\mathcal{B}} \overset{ind.\ hyp.}{=} \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}}\} \cap \{nat_R \circ g \mid g \in \psi^{\mathcal{A}}\} \\ &= \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}} \cap \psi^{\mathcal{A}}\} = \{nat_R \circ g \mid g \in (\varphi \wedge \psi)^{\mathcal{A}}\}, \\ (\forall x\varphi)^{\mathcal{B}} &= \{f \in (A/R)^V \mid \forall\ [a]_R \in (A/R)_e : f[[a]_R/x] \in \varphi^{\mathcal{B}}\} \\ &= \{nat_R \circ g \mid g \in A^V,\ \forall\ [a]_R \in (A/R)_e : (nat_R \circ g)[[a]_R/x] \in \varphi^{\mathcal{B}}\} \\ &= \{nat_R \circ g \mid g \in A^V,\ \forall\ a \in A_e : nat_R \circ g[a/x] \in \varphi^{\mathcal{B}}\} \\ &\overset{ind.\ hyp.}{=} \{nat_R \circ g \mid g \in A^V,\ \forall\ a \in A_e : g[a/x] \in \varphi^{\mathcal{A}}\} \\ &= \{nat_R \circ g \mid g \in (\forall x\varphi)^{\mathcal{A}}\}. \end{aligned}$$

Proof of (3). Suppose that $\mathcal{A}$ satisfies φ. Then $\varphi^{\mathcal{A}} = A^V$ and thus by (2),

$$\varphi^{\mathcal{B}} = \{nat_R \circ g \mid g \in \varphi^{\mathcal{A}}\} = \{nat_R \circ g \mid g \in A^V\} = (A/R)^V,$$

i.e., $\mathcal{B}$ satisfies φ. ❑

13.6 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.

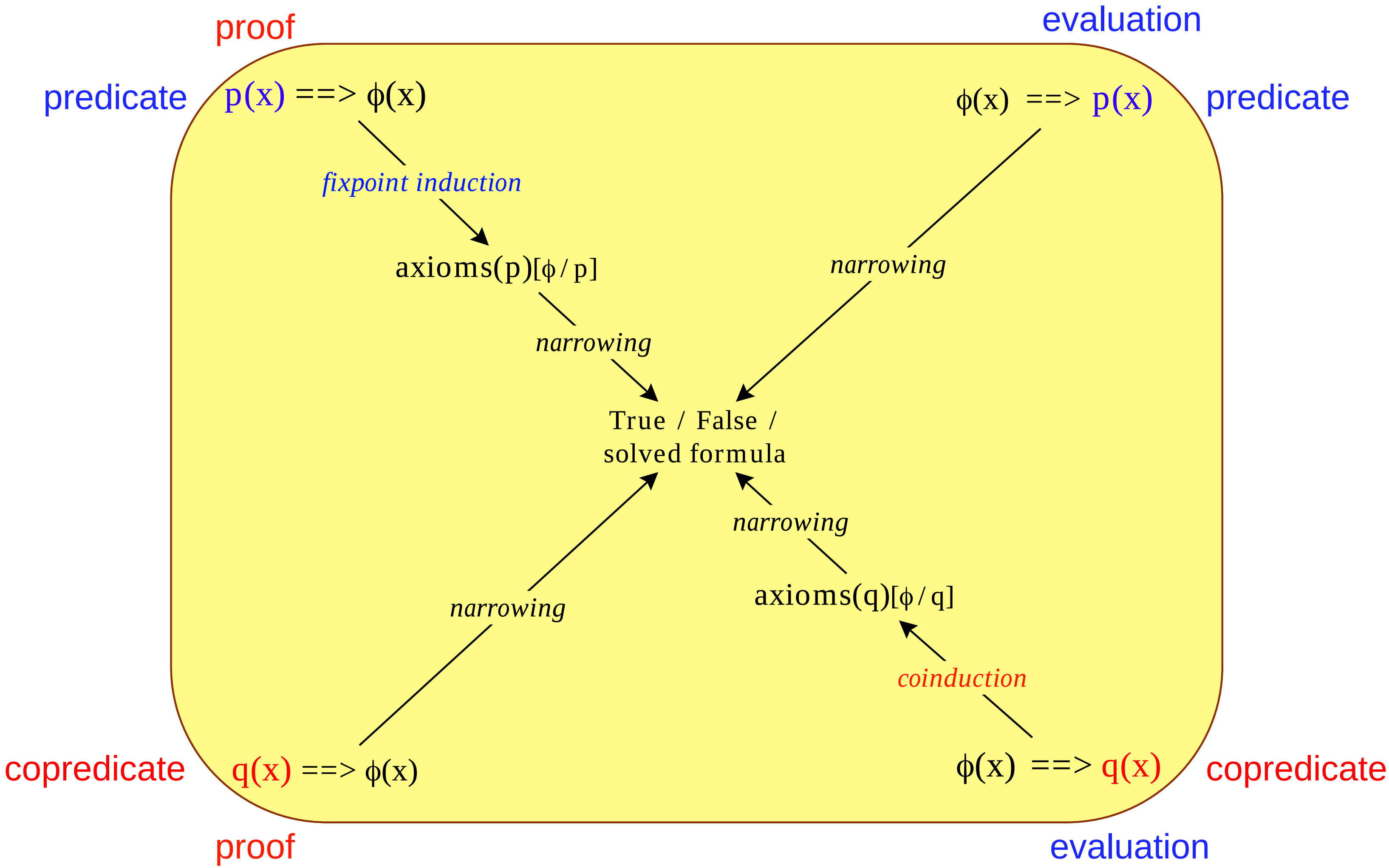
(Co)Induction applies conjectures to axioms.
The proof proceeds by transforming the modified axioms.

Resolution upon a predicate p is a rule for evaluating p.

Induction upon a predicate p is a rule for verifying p.

Coresolution upon a predicate q is a rule for verifying q.

Coinduction upon a predicate q is a rule for evaluating q.


proof
evaluation
predicate
p(x) ==> ϕ(x)
ϕ(x) ==> p(x)
predicate
fixpoint induction
narrowing
axioms(p)[ϕ/p]
narrowing
True / False /
solved formula
narrowing
narrowing
axioms(q)[ϕ/q]
coinduction
copredicate
q(x) ==> ϕ(x)
ϕ(x) ==> q(x)
copredicate
proof
evaluation

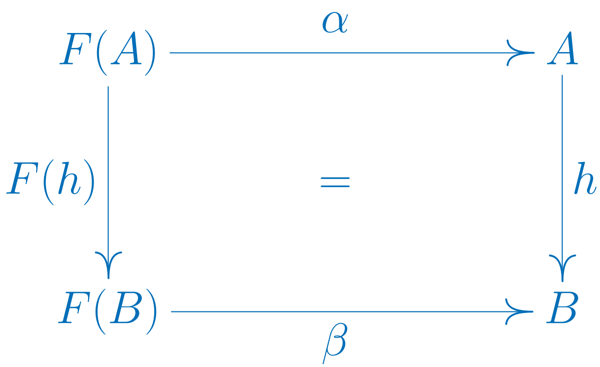
14 F-algebras and -coalgebras

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 $\alpha : F(A) \to A$.

Alg_F denotes the category of F-algebras and the following $\mathcal{K}$-morphisms:

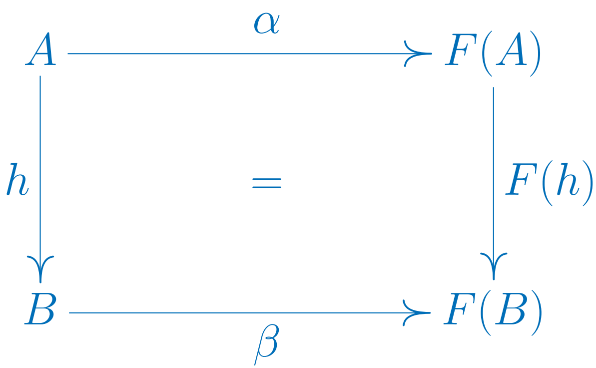
An Alg_F**-morphism** h from an F-algebra $\alpha\!:\!F(A) \to A$ to an F-algebra $\beta\!:\!F(B) \to B$ is a $\mathcal{K}$-morphism $h : A \to B$ such that $h \circ \alpha = \beta \circ F(h)$, i.e., the following diagram commutes:



An F**-coalgebra** or F**-codynamics** [17] is a $\mathcal{K}$-morphism $\alpha : A \to F(A)$.

coAlg$_F$ denotes the category of F-coalgebras and the following $\mathcal{K}$-morphisms:

A *coAlg*$_F$**-morphism** h from an F-coalgebra $\alpha : A \to F(A)$ to an F-coalgebra $\beta : B \to F(B)$ is a $\mathcal{K}$-morphism $h : A \to B$ such that $F(h) \circ \alpha = \beta \circ h$, i.e., the following diagram commutes:



A $\mathcal{K}$-object A is a **fixpoint of** F if $F(A) \cong A$.

Lemma 14.1 (Lambek's Lemma; [97], Lemma 2.2; [23], Prop. 5.12; [15], section 2; [156], Thm. 9.1)

(1) Suppose that Alg_F has an initial object $\alpha : F(A) \to A$.

α is iso and thus A is a fixpoint of F.

(2) Suppose that $coAlg_F$ has a final object $\beta : A \to F(A)$.

β is iso and thus A is a fixpoint of F.

Proof. (1) Since α is initial, there is an Alg_F-morphism $h : A \to F(A)$ from α to $F(\alpha)$. Hence $\alpha \circ h$ is an Alg_F-morphism from α to α:

$$\alpha \circ h \circ \alpha = \alpha \circ F(\alpha) \circ F(h) = \alpha \circ F(\alpha \circ h).$$

id_A is also an Alg_F-morphism from α to α:

$$id_A \circ \alpha = \alpha = \alpha \circ id_{F(A)} = \alpha \circ F(id_A).$$

Hence (3) $\alpha \circ h = id_A$ because α is initial in Alg_F. Since h is an Alg_F-morphism,

$$h \circ \alpha = F(\alpha) \circ F(h) = F(\alpha \circ h) = F(id_A) = id_{F(A)}. \tag{4}$$

By (3) and (4), α is an isomorphism.

(2) Analogously. ❑

Let $\alpha : F(A) \to A$ be an F-algebra and $ini_F : F(\mu F) \to \mu F$ be initial in Alg_F.

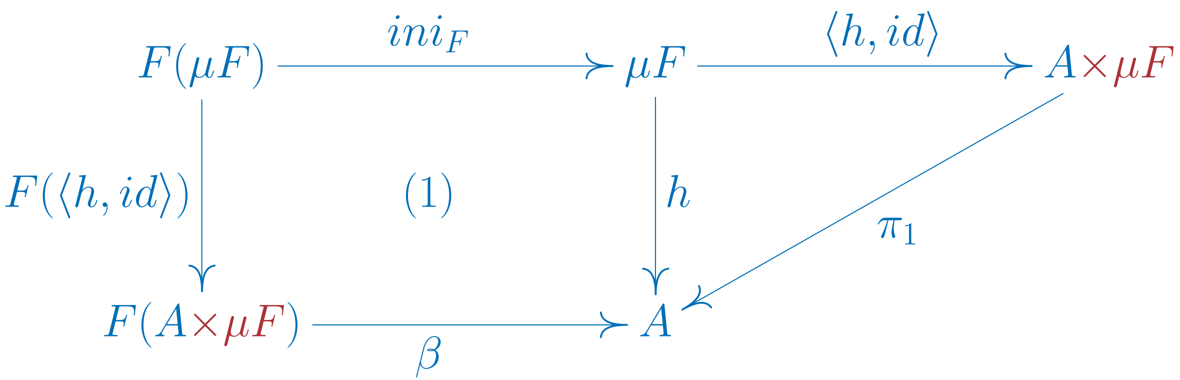
The unique Alg_F-morphism from ini_F to α is called a **catamorphism** [107, 180], **reachability map** [17] or a function **defined by recursion** and denoted by $fold^\alpha$ or $(|\alpha|) : \mu F \to A$.

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 16.1 and section 16.3).

Lemma 14.2 Let $\beta : F(A\times\mu F) \to A$ be a $\mathcal{K}$-morphism and γ be the F-algebra

$$\langle \beta, ini_F \circ F(\pi_2)\rangle : F(A \times \mu F) \to A \times \mu F.$$

There is a unique $\mathcal{K}$-morphism $h : \mu F \to A$ such that (1) commutes:



Proof. Suppose that (1) holds true. Then

$$\langle h, id\rangle \circ ini_F \overset{(6)\ on\ p.\ 16}{=} \langle h \circ ini_F, id \circ ini_F\rangle$$
$$\overset{id\ is\ Alg_F-morph.}{=} \langle h \circ ini_F, ini_F \circ F(id)\rangle = \langle h \circ ini_F, ini_F \circ F(\pi_2 \circ \langle h, id\rangle)\rangle$$
$$= \langle h \circ ini_F, ini_F \circ F(\pi_2) \circ F(\langle h, id\rangle)\rangle$$
$$\overset{(1)}{=} \langle \beta \circ F(\langle h, id\rangle), ini_F \circ F(\pi_2) \circ F(\langle h, id\rangle)\rangle$$
$$\overset{(6)\ on\ p.\ 16}{=} \langle \beta, ini_F \circ F(\pi_2)\rangle \circ F(\langle h, id\rangle) = \gamma \circ F(\langle h, id\rangle).$$

Hence $\langle h, id\rangle : \mu F \to A \times \mu F$ is an Alg_F-morphism from ini_F to γ. Given a further $\mathcal{K}$-morphism $h : \mu F \to A$ 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 γ, $h = \pi_1 \circ \langle h, id\rangle = \langle h', id\rangle = h'$. ❏

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 19).

As to primitive recursion, the right-adjointness of products to diagonals provides the reduction: In terms of Theorem 25.1, $h : \mu F \to A$ is a paramorphism iff

$$(h, id) : (\mu F, \mu F) \to (A, \mu F)$$

is $(\Delta, \_ \times \mu F)$-recursive, i.e., iff $(h, id)^\# = \langle h, id \rangle$ is an Alg_F-morphism.

In section 16.3, many recursion schemas and their reduction to (1) are exemplified. For instance, the equations given there for the factorial function yield a paramorphism.

Let $\alpha : A \to F(A)$ be an F-coalgebra and $fin_F : \nu F \to F(\nu F)$ be final in $coAlg_F$.

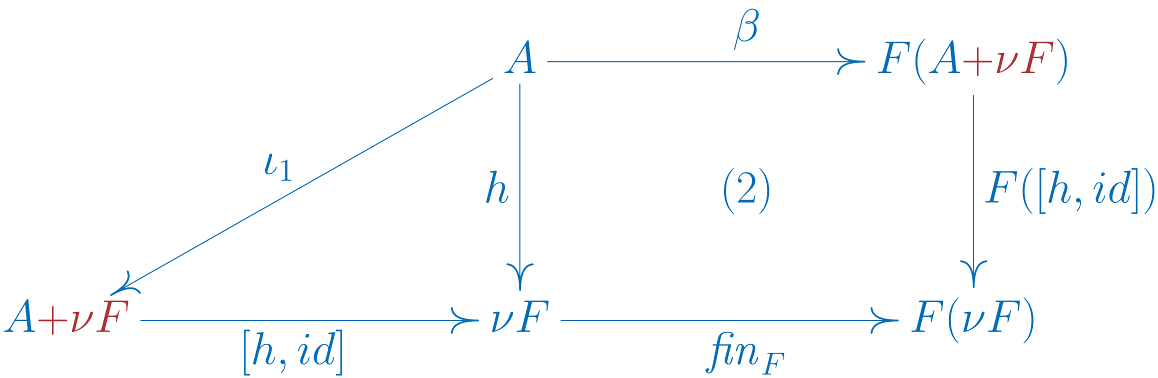
The unique $coAlg_F$-morphism from α to β is called an **anamorphism** [107, 180] or **observability map** [17] and denoted by $unfold^\alpha$ or $|(\alpha)| : A \to \nu F$.

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 16.2 and section 16.4).

Lemma 14.3 Let $\beta : A \to F(A+\nu F)$ be a $\mathcal{K}$-morphism and γ be the F-coalgebra

$$[\beta, F(\iota_2) \circ \mathit{fin}_F] : A + \nu F \to F(A + \nu F).$$

There is a unique $\mathcal{K}$-morphism $h : A \to \nu F$ such that (2) commutes:



Proof. Analogously to the proof of Theorem 14.2. ❑

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 19).

As to primitive corecursion, the left-adjointness of coproducts (sums) to diagonals provides the reduction: In terms of Theorem 25.2, h is an apomorphism iff

$$(h, id) : (A, \nu F) \to (\nu F, \nu F)$$

is $(\_ + \mu F, \Delta)$-corecursive, i.e., iff $(h, id)^* = [h, id]$ is an Alg_F-morphism.

In sections 16.4 and 16.5, 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 $ini_F : F(\mu F) \to \mu F$ be initial in Alg_F and $fin_F : \nu F \to F(\nu F)$ be final in $coAlg_F$ such that μF embedded in νF. Moreover, let Fin_F be the subcategory of $coAlg_F$ that consists of all F-coalgebras $\alpha : A \to F(A)$ such that $unfold^\alpha : A \to \nu F$ factors through μF.

Let $\alpha : F(A) \to A$ be an F-algebra and $\beta : B \to F(B)$ be an F-coalgebra. A $\mathcal{K}$-morphism $h : B \to A$ is a **hylo(morphism)** w.r.t. (α, β) if

$$h = \alpha \circ F(h) \circ \beta \tag{3}$$

(see [76], section 3). If h is unique with (3), we write $[|\alpha, \beta|]$ for h (see [107, 45]).

β is **recursive** [9, 38] if for every F-algebra $\alpha : F(A) \to A$ there is a unique hylo w.r.t. (α, β).

α is **corecursive** [8, 39] if for every F-coalgebra $\beta : B \to F(B)$ there is a unique hylo w.r.t. (α, β).

Hence

- by Lemma 14.1 (1) and Lemma 4.2 (1), the inverse of an initial F-algebra $ini_F : F(\mu F) \to \mu F$ is a recursive F-coalgebra with hylo

$$[|\alpha, ini_F^{-1}|] = (|\alpha|) : \mu F \to A; \tag{4}$$

- by Lemma 14.1 (2) and Lemma 4.2 (2), the inverse of a final F-coalgebra $fin_F : \nu F \to F(\nu F)$ is a corecursive F-algebra with hylo

$$[|fin_F^{-1}, \beta|] = |(\beta)| : B \to \nu F. \tag{5}$$

Lemma 14.4 (Hylo-Compose [52, 45])

Let $\alpha : F(A) \to A$ and $\beta' : F(B) \to B$ be F-algebras and $\beta : B \to F(B), \gamma : C \to F(C)$ be F-coalgebras such that $\beta \circ \beta' = id_{F(B)}$ and there are a hylo $g : B \to A$ w.r.t. (α, β) and a hylo $h : C \to B$ w.r.t. (β', γ).

(6) $g \circ h : C \to A$ is a hylo w.r.t. (α, γ).

(7) Let $in_F : F(B) \to B$ be initial in Alg_F, $fin_F : B \to F(B)$ 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. (α, γ).

Proof. (6):

$$g \circ h \stackrel{g\ hylo}{=} \alpha \circ F(g) \circ \beta \circ h \stackrel{h\ hylo}{=} \alpha \circ F(g) \circ \beta \circ \beta' \circ F(h) \circ \gamma$$
$$\stackrel{\beta \circ \beta' = id}{=} \alpha \circ F(g) \circ F(h) \circ \gamma = \alpha \circ F(g \circ h) \circ \gamma.$$

(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. (α, γ). ❏

Let $\alpha' : F(A) \times B \to A$ be a $\mathcal{K}$-morphism and $\beta : B \to F(B)$ be an F-coalgebra. A $\mathcal{K}$-morphism $h : B \to A$ is a **para-hylo**(**morphism**) w.r.t. (α', β) if

$$h = \alpha' \circ \langle F(h) \circ \beta, id_B \rangle \tag{8}$$

(see [76], section 5).

β is **parametrically recursive** [9] if for every $\mathcal{K}$-morphism $\alpha' : F(A) \times B \to A$ there is a unique para-hylo w.r.t. (α', β).

Since $\langle F(h) \circ \beta, id_B \rangle = (F(h) \times id_B) \circ \langle \beta, id_B \rangle$, $\alpha : F(A) \to A$ satisfies (5) iff

$$\alpha' = \alpha \circ \pi_1 : F(A) \times B \to A$$

solves (10), i.e., every parametrically recursive F-coalgebra is recursive.

Given a destructive signature Σ, the converse holds true as well: all recursive H_Σ-coalgebras (see chapter 15) are parametrically recursive ([9], Theorem 3.8).

Let $\alpha : F(A) \to A$ be an F-algebra and $\beta' : B \to F(B) + A$ be a $\mathcal{K}$-morphism. A $\mathcal{K}$-morphism $h : B \to A$ is an **apo-hylo**(**morphism**) w.r.t. (α, β') if

$$h = [\alpha \circ F(h), id_A] \circ \beta' \tag{9}$$

(see [76], section 5).

α is **parametrically corecursive** or **completely iterative** [8, 39] if for every $\mathcal{K}$-morphism $\beta' : B \to F(B) + A$ there is a unique apo-hylo w.r.t. (α, β').

Since $[\alpha \circ F(h), id_A] = [\alpha, id_A] \circ (F(h) + id_A)$, $\beta : B \to F(B)$ satisfies (5) iff

$$\beta' = \beta \circ \iota_1 : B \to F(B) + A$$

solves (9), i.e., every parametrically corecursive F-algebra is corecursive.

According to [8], section 9, the converse does not hold true.

14.1 Invariants and congruences

(see [81], Defs. 3.1.1, 3.1.2, 6.1.1, 6.2.1; [94], Def. 2.5)

Let $F : Set \to Set$ be a functor, A be a set, $B \subseteq A$ and $R \subseteq A^2$.

$$Pred(F)(B) =_{def} \{F(inc_B)(c) \mid c \in F(B)\} \subseteq F(A), \qquad \text{(predicate lifting)}$$

$$Rel(F)(R) =_{def} \{(F(\pi_1)(c), F(\pi_2)(c)) \mid c \in F(R)\} \subseteq F(A)^2. \qquad \text{(relation lifting)}$$

Let $\alpha : F(A) \to A$ be an F-algebra,

$$\begin{aligned}
\Phi_\alpha : \mathcal{P}(A) &\to \mathcal{P}(A) \\
B &\mapsto \{\alpha(c) \mid c \in \mathit{Pred}(F)(B)\}, \\
\Psi_\alpha : \mathcal{P}(A^2) &\to \mathcal{P}(A^2) \\
R &\mapsto \{(\alpha(c), \alpha(d)) \mid (c, d) \in \mathit{Rel}(F)(R)\}.
\end{aligned}$$

B is an **invariant** of α if for all $c \in \mathit{Pred}(F)(B)$, $\alpha(c) \in B$, or, equivalently, if B is Φ_α-closed.

R is a **bisimulation** on α if for all $(c, d) \in \mathit{Rel}(F)(R)(B)$, $(\alpha(c), \alpha(d)) \in R$, or, equivalently, if R is Ψ_α-closed.

By Theorem 3.9 (1),

$$\begin{aligned}
\mathit{lfp}(\Phi_\alpha) &= \bigcap\{B \subseteq A \mid B \text{ is } \Phi_\alpha\text{-closed}\}, \\
\mathit{lfp}(\Psi_\alpha) &= \bigcap\{R \subseteq A^2 \mid R \text{ is } \Psi_\alpha\text{-closed}\}.
\end{aligned}$$

Hence $\mathit{lfp}(\Phi_\alpha)$ is the least invariant of α and $\mathit{lfp}(\Psi_\alpha)$ is the least bisimulation on α.

Let $\beta : A \to F(A)$ be an F-coalgebra,

$$\begin{aligned} \Phi_\beta : \mathcal{P}(A) &\to \mathcal{P}(A) \\ B &\mapsto \{a \in A \mid \beta(a) \in \mathit{Pred}(F)(B)\}, \\ \Psi_\beta : \mathcal{P}(A^2) &\to \mathcal{P}(A^2) \\ R &\mapsto \{(a, b) \mid (\beta(a), \beta(b)) \in \mathit{Rel}(F)(R)\}. \end{aligned}$$

B is an **invariant** of β if for all $a \in B$, $\beta(a) \in \mathit{Pred}(F)(B)$, or, equivalently, if B is Φ_β-dense.

R is a **bisimulation** on β if for all $(a, b) \in R$, $(\beta(a), \beta(b)) \in \mathit{Rel}(F)(R)$, or, equivalently, if R is Ψ_β-dense.

By Theorem 3.9 (5),

$$\begin{aligned} \mathit{gfp}(\Phi_\beta) &= \bigcup\{B \subseteq A \mid B \text{ is } \Phi_\beta\text{-dense}\}, \\ \mathit{gfp}(\Psi_\beta) &= \bigcup\{R \subseteq A^2 \mid R \text{ is } \Psi_\beta\text{-dense}\}. \end{aligned}$$

Hence $\mathit{gfp}(\Phi_\beta)$ is the greatest invariant of β and $\mathit{gfp}(\Psi_\beta)$ is the greatest bisimulation on β. Moreover, by Theorem 9.6 (2), greatest bisimulations are equivalence relations.

Therefore, $gfp(\Psi_\beta)$ is also the greatest congruence on β.

14.2 Complete categories and continuous functors

Let $\mathcal{K}$ and $\mathcal{L}$ be λ-complete (see section 6.1). A functor $F : \mathcal{K} \to \mathcal{L}$ is **λ-continuous** if for all λ-cochains $\mathcal{D}$ of $\mathcal{K}$, F preserves the limit $\{\nu_i : C \to \mathcal{D}(i) \mid i < \lambda\}$ of $\mathcal{D}$, i.e., $\{F(\nu_i) \mid i < \lambda\}$ is the limit of $F \circ \mathcal{D}$.

Let $\mathcal{K}$ and $\mathcal{L}$ be λ-cocomplete (see section 6.2). A functor $F : \mathcal{K} \to \mathcal{L}$ is **λ-cocontinuous** if for all λ-chains $\mathcal{D}$ of $\mathcal{K}$, F preserves the colimit $\{\mu_i : \mathcal{D}(i) \to C \mid i < \lambda\}$ of $\mathcal{D}$, i.e., $\{F(\mu_i) \mid i < \lambda\}$ is the colimit of $F \circ \mathcal{D}$.

CPO^E denotes the category of ω-CPOs as objects and pairs

$$(f : A \to B, g : B \to A)$$

of ω-continuous functions with $g \circ f = id_A$ and $f \circ g \leq id_B$ as morphisms.

Theorem 14.5 (see, e.g., [125], section 11.3)

All endofunctors on CPO^E built up from identity and constant functors, coproducts, finite products and hom-functors are cocontinuous. ❑

$e \in \mathcal{T}_{po}(S)$ is **strongly polynomial** if e contains only product types with a *finite* set of indices.

Let κ be a cardinal number. $e \in \mathcal{T}_{po}(S)$ is κ**-polynomial** if e does not contain a product type whose set of indices has a cardinality greater than κ.

Theorem 14.6

(1) For all strongly polynomial types e over S, F_e is ω-continuous.

Let κ be a cardinal number and λ 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 κ-polynomial types e over S, F_e is λ-cocontinuous.

(3) For all κ-polynomial types e over S, F_e is λ-continuous.

Proof. By [17], Thms. 1 and 4, or [21], Prop. 2.2 (1) and (2), permutative and constant functors are ω-continuous and ω-cocontinuous, ω-continuous or λ-cocontinuous functors are closed under coproducts, ω-continuous functors are closed under products (and thus under exponentiation; see [156], Thm. 10.1) and λ-cocontinuous functors are closed under finite products.

By [21], Prop. 2.2 (3), ω-continuous or λ-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 $\mathbb{N}_\omega^A \cong A^*/{=_{bag}}$, the functors $\_^*$ and $\mathbb{N}_\omega^-$ are ω-continuous and ω-cocontinuous (see [10], Exs. 2.3.14/15).

By [10], Ex. 2.2.13, $\mathcal{P}_\omega$ is ω-cocontinuous. For a proof of the fact that $\mathcal{P}_\omega$ is not ω-continuous, see [10], Ex. 2.3.11. $\mathcal{P}_\omega(A)$ is a quotient of A^*, but not a finite one: $\mathcal{P}_\omega(A) \cong A^*/{=_{set}}$ (see chapter 2).

By [10], Thm. 4.1.12, λ-cocontinuous functors are closed under products whose index sets have cardinalities less than λ and thus under exponentiation by exponents with a cardinality less than λ. Moreover, ω-continuous or λ-cocontinuous functors are closed under sequential composition. ❏

14.3 Initial F-algebras and final F-coalgebras

Theorem 14.7 (For $\lambda = \omega$, see [15], section 2; [100], Thm. 2.1; for any λ, see [4], [6], Thm. 3.19, or [10], Cor. 4.1.5.)

Let λ be an infinite cardinal, Ini be initial in $\mathcal{K}$ and $\mathcal{K}$ be κ-cocomplete for all $\kappa \leq \lambda$.

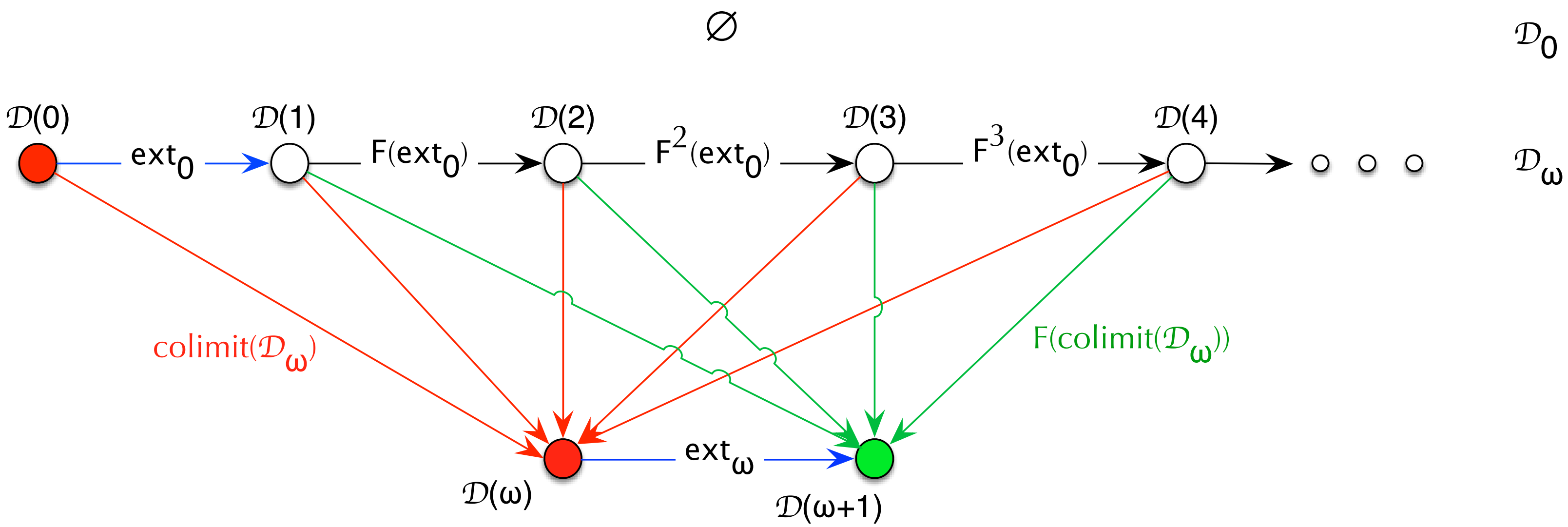
Given an endofunctor F on $\mathcal{K}$, define a λ-chain $\mathcal{D}$ of $\mathcal{K}$ as follows:

$$\begin{array}{rcll} \mathcal{D}(0) & = & Ini, & \\ \mathcal{D}(0,1) & = & ext_0 : \mathcal{D}(0) \to \mathcal{D}(1), & \\ \mathcal{D}(k+1) & = & F(\mathcal{D}(k)) & \text{for all } k < \lambda, \\ \mathcal{D}(i+1,k+1) & = & F(\mathcal{D}(i,k)) & \text{for all } i < k < \lambda, \\ \mathcal{D}(i,k) & = & \mu_{i,k} : \mathcal{D}(i) \to \mathcal{D}(k) & \text{for all limit ordinals } k < \lambda \text{ and all } i < k, \\ \mathcal{D}(k,k+1) & = & ext_k : \mathcal{D}(k) \to \mathcal{D}(k+1) & \text{for all limit ordinals } k < \lambda \end{array}$$

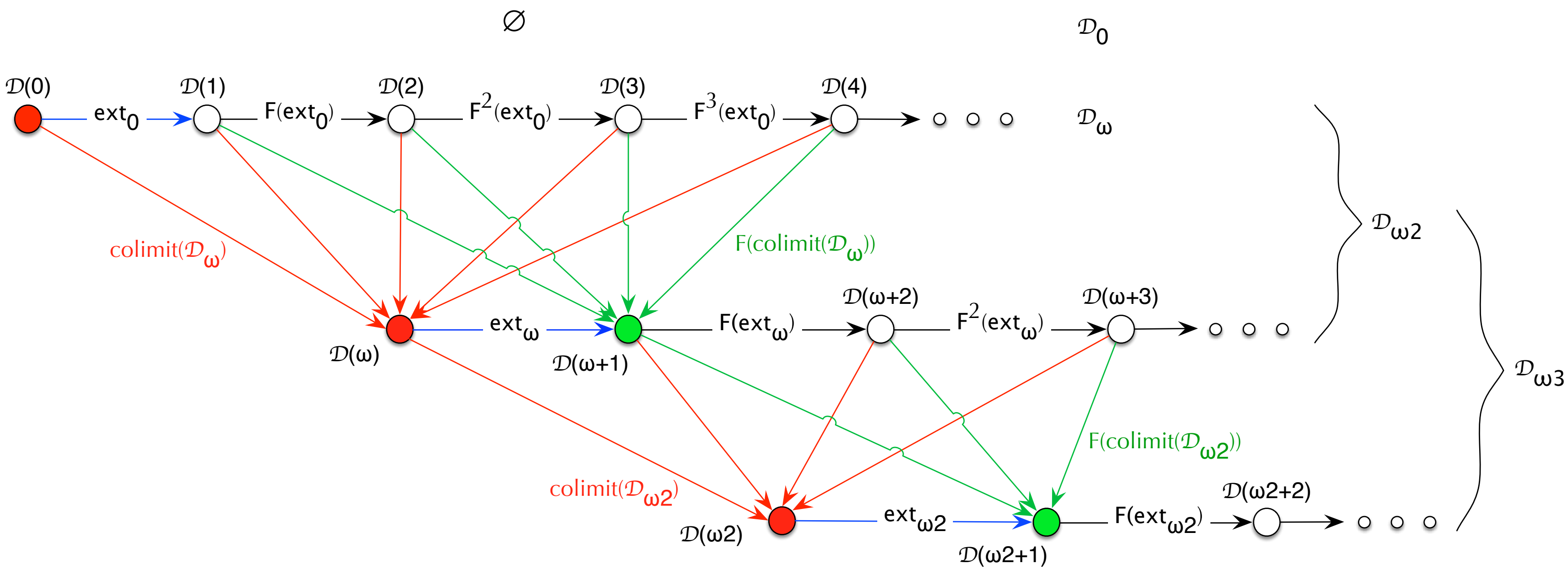
where ext_0 is the unique $\mathcal{K}$-morphism from Ini to $F(Ini)$ and for all limit ordinals $k < \lambda$, $\gamma_k = \{\mu_{i,k} \mid i < k\}$ is the colimit of the greatest subdiagram $\mathcal{D}_k : \mathbb{O}_k \to \mathcal{K}$ of $\mathcal{D}$ and ext_k is the unique $\mathcal{K}$-morphism from $\mathcal{D}(k)$ to $F(\mathcal{D}(k))$ such that for all $i < k$,

$$ext_k \circ \mu_{i+1,k} = F(\mu_{i,k}) : \mathcal{D}(i+1) \to \mathcal{D}(k+1).$$

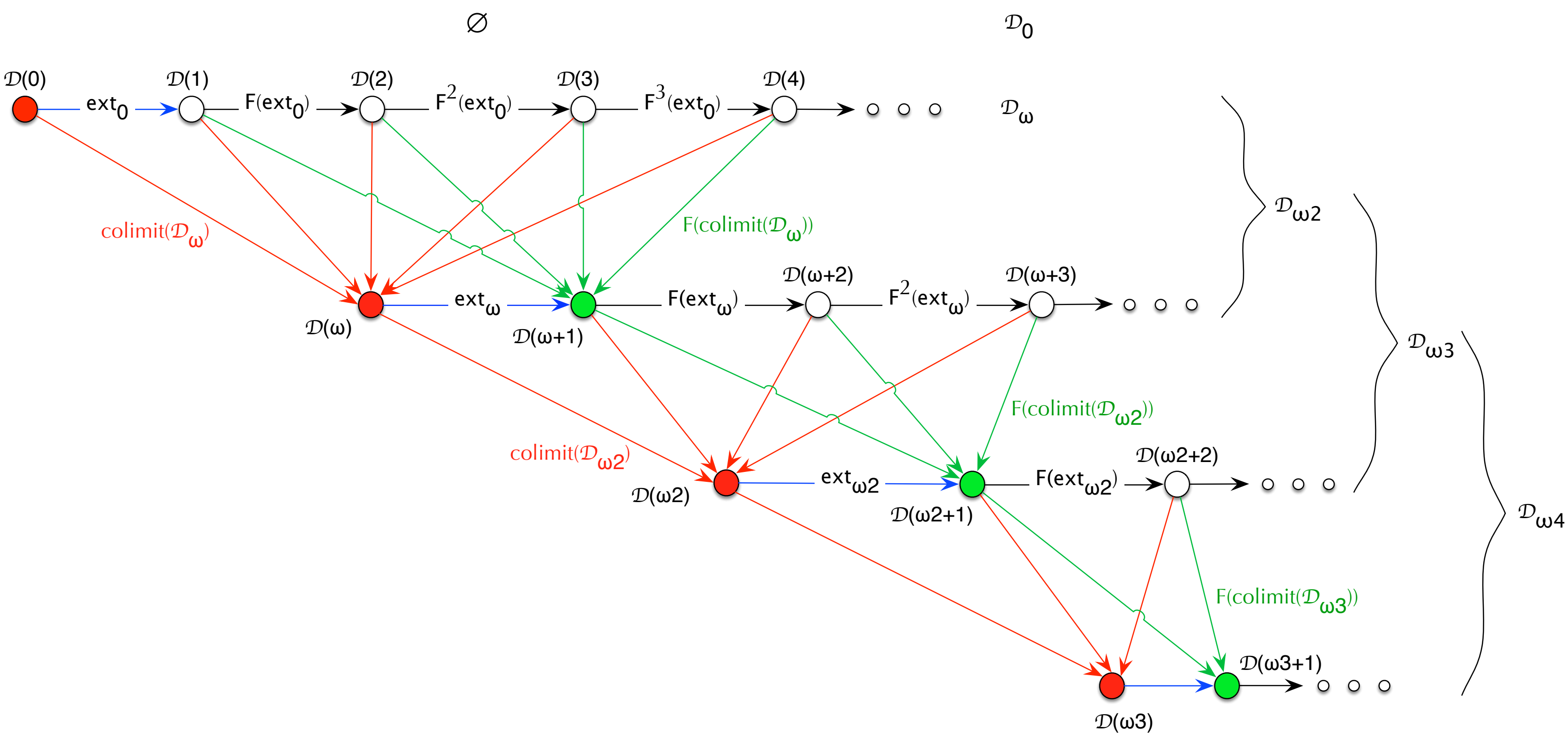
ext_k exists because $\{F(\mu_{i,k}) \mid i < k\}$ is a cocone of $F \circ \mathcal{D}_k$ and $\gamma_k \setminus \{\mu_{0,k}\}$ is the colimit of $F \circ \mathcal{D}_k$.



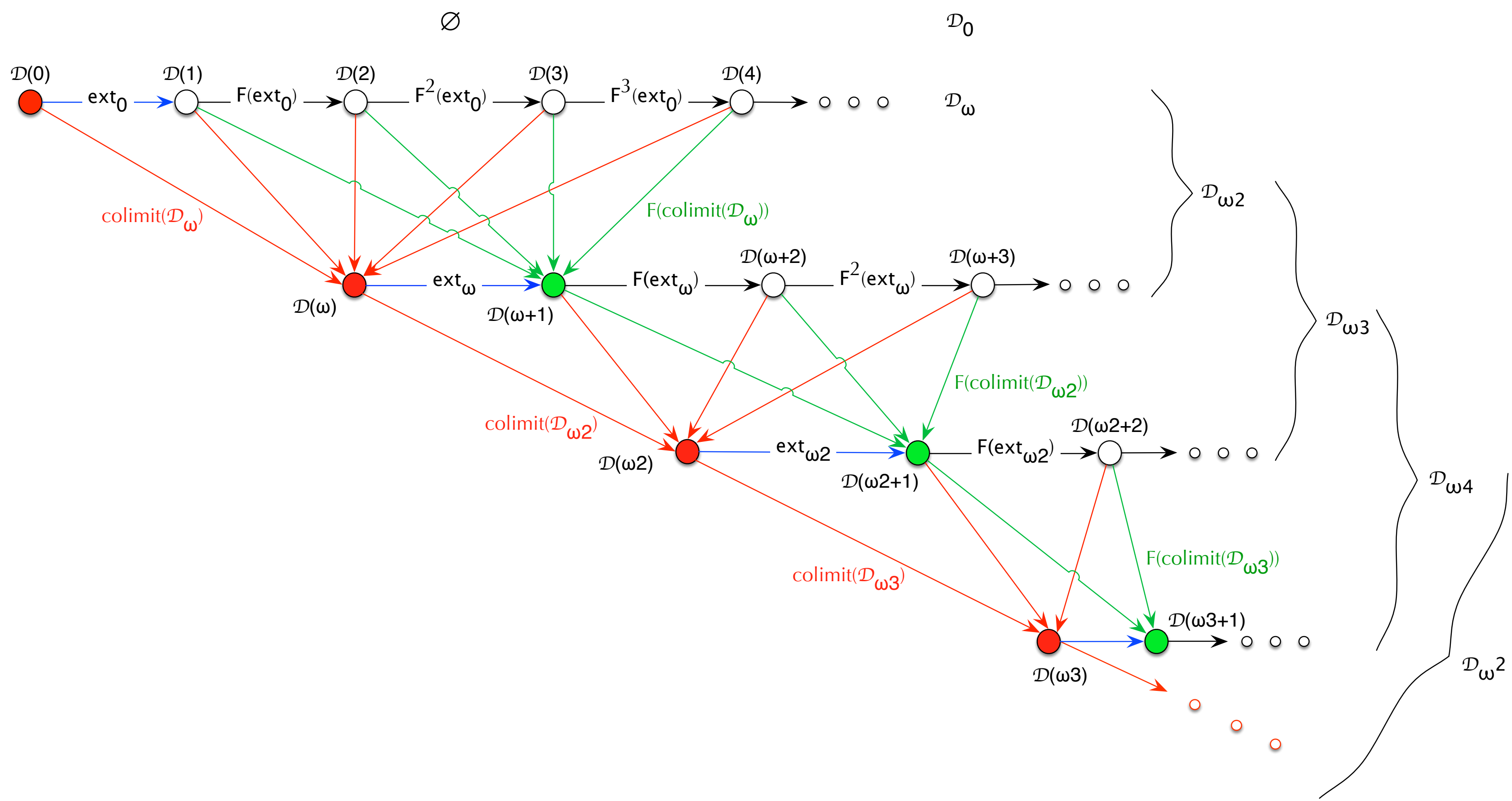
The $\omega + 2$-chain of $\mathcal{K}$ induced by the initial object $\mathcal{D}(0)$ of $\mathcal{K}$



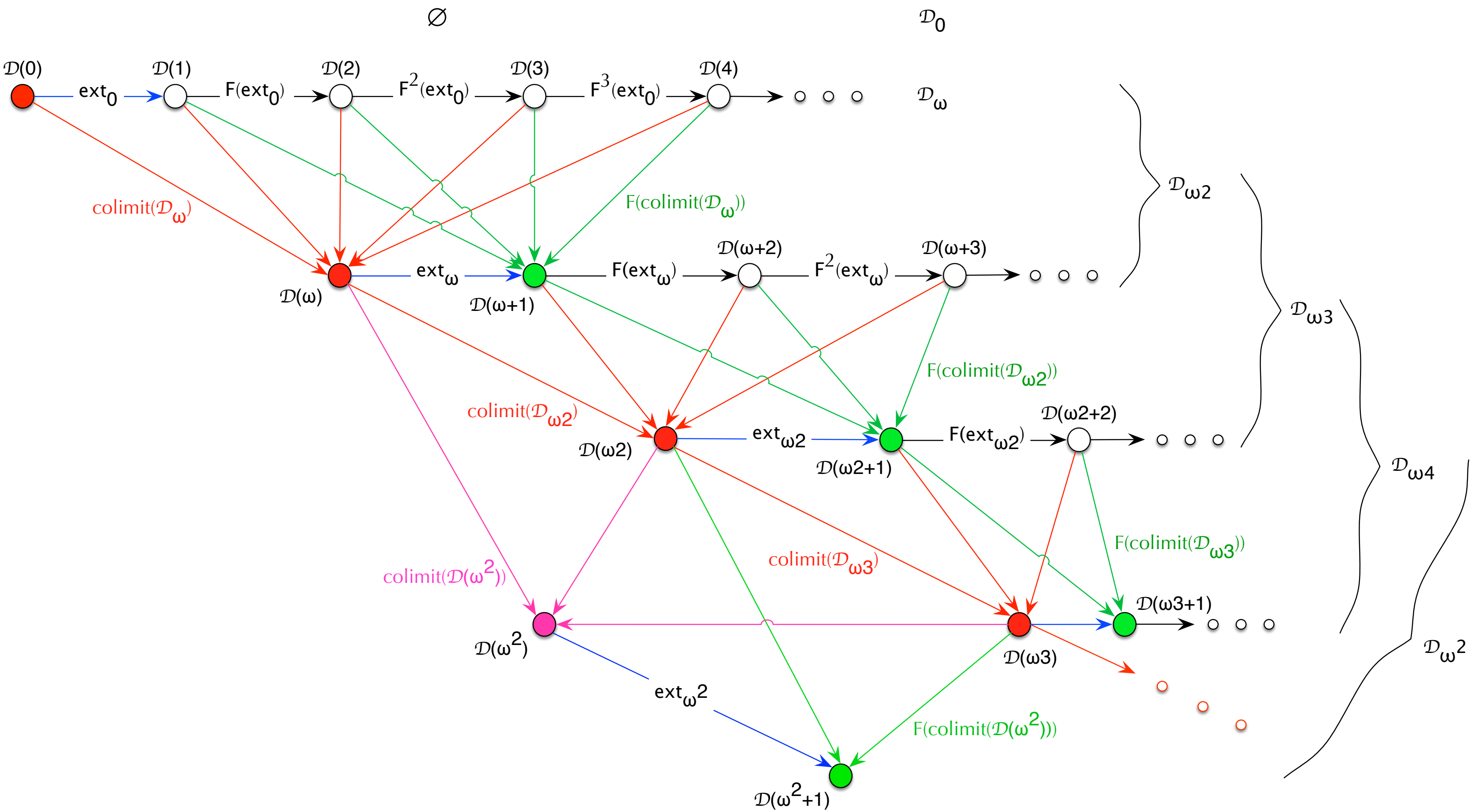
The $\omega 3$-chain of $\mathcal{K}$ induced by the initial object $\mathcal{D}(0)$ of $\mathcal{K}$



The $\omega 4$-chain of $\mathcal{K}$ induced by the initial object $\mathcal{D}(0)$ of $\mathcal{K}$



The ω^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 λ-cocontinuous,

$$\mu = \{\mu_i : \mathcal{D}(i) \to \mathcal{D}(\lambda) \mid i < \lambda\}$$

be the colimit of $\mathcal{D}$. Then

$$F(\mu) = \{F(\mu_i) : F(\mathcal{D}(i)) \to F(\mathcal{D}(\lambda)) \mid i < \lambda\}$$

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 $ext : F(\mathcal{D}(\lambda)) \to \mathcal{D}(\lambda)$—and thus an F-algebra—such that for all $i < \lambda$,

$$ext \circ F(\mu_i) = \mu_{i+1} : \mathcal{D}(i+1) \to \mathcal{D}(\lambda).$$

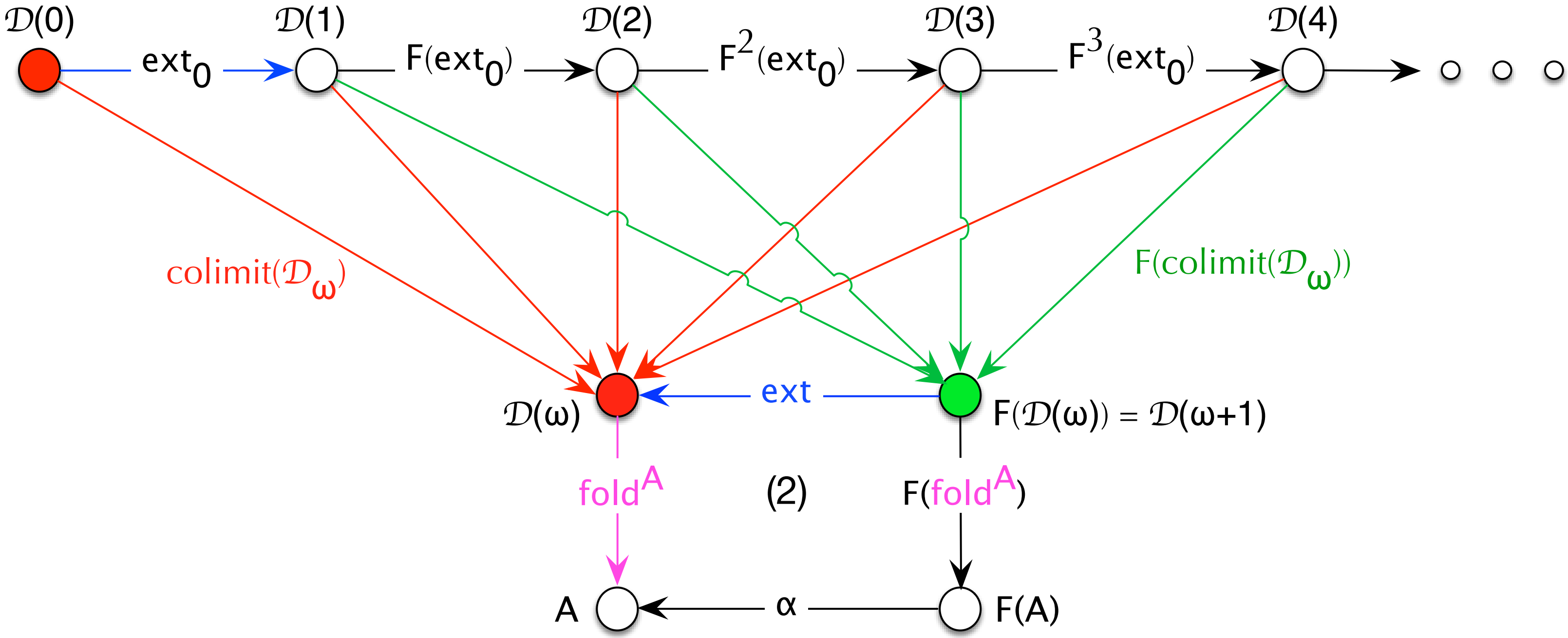
ext is initial in Alg_F and thus, by Lemma 14.1 (1), $F(\mathcal{D}(\lambda)) \cong \mathcal{D}(\lambda)$.

Proof. Let $\alpha : F(A) \to A$ 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

$$\nu = \{\nu_i : \mathcal{D}(i) \to A \mid i < \lambda\}$$

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 $fold^A : \mathcal{D}(\lambda) \to A$ with $fold^A \circ \mu_i = \nu_i$ for all $i < \lambda$. Therefore,

$$\begin{aligned} & fold^A \circ ext \circ F(\mu_i) = fold^A \circ \mu_{i+1} = \nu_{i+1} = \alpha \circ F(\nu_i) = \alpha \circ F(fold^A \circ \mu_i) \\ & = \alpha \circ F(fold^A) \circ F(\mu_i). \qquad (1) \end{aligned}$$



The initial F-algebra 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 $h : F(\mathcal{D}(\lambda)) \to A$ with $h \circ F(\mu_i) = \nu_{i+1}$ for all $i < \lambda$. Hence (1) implies

$$fold^A \circ ext = \alpha \circ F(fold^A), \tag{2}$$

i.e., $fold^A$ is an Alg_F-morphism from ext to α.

It remains to show that $fold^A$ is the only Alg_F-morphism from ext to α.

Let $\theta : \mathcal{D}(\lambda) \to A$ be an Alg_F-morphism from ext to α, i.e.,

$$\theta \circ ext = \alpha \circ F(\theta). \tag{3}$$

Suppose that for all $i < \lambda$,

$$\theta \circ \mu_i = \nu_i : \mathcal{D}(i) \to A. \tag{4}$$

Since $fold^A \circ \mu_i = \nu_i$ and there is only one $\mathcal{K}$-morphism $h : \mathcal{D}(\lambda) \to A$ with $h \circ \mu_i = \nu_i$, we conclude $\theta = fold^A$.

It remains to show (4) by transfinite induction on i.

Since $\mathcal{D}(0) = Ini$ 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

$$\begin{aligned}\theta \circ \mu_k = \theta \circ \mu_{i+1} = \theta \circ ext \circ F(\mu_i) &\overset{(3)}{=} \alpha \circ F(\theta) \circ F(\mu_i) = \alpha \circ F(\theta \circ \mu_i) \\ &\overset{ind.\ hyp.}{=} \alpha \circ F(\nu_i) = \nu_{i+1} = \nu_k.\end{aligned}$$

Let k be a limit ordinal. Since μ and ν 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,

$$\theta \circ \mu_k \circ \mu_{i,k} = \theta \circ \mu_i = \nu_i = \nu_k \circ \mu_{i,k}. \tag{5}$$

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 $h : \mathcal{D}(k) \to A$ 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. ❏

Theorem 14.8

Let λ be an infinite cardinal, Fin be final in $\mathcal{K}$ and $\mathcal{K}$ be κ-complete for all $\kappa \leq \lambda$.

Given an endofunctor F on $\mathcal{K}$, define a λ-cochain $\mathcal{D}$ of $\mathcal{K}$ as follows:

$$\begin{aligned}
\mathcal{D}(0) &= Fin, \\
\mathcal{D}(1,0) &= ext_0 : \mathcal{D}(1) \to \mathcal{D}(0), \\
\mathcal{D}(k+1) &= F(\mathcal{D}(k)) && \text{for all } k < \lambda, \\
\mathcal{D}(k+1,i+1) &= F(\mathcal{D}(k,i)) && \text{for all } i < k < \lambda, \\
\mathcal{D}(k,i) &= \mu_{k,i} : \mathcal{D}(k) \to \mathcal{D}(i) && \text{for all limit ordinals } k < \lambda \text{ and all } i < k, \\
\mathcal{D}(k+1,k) &= ext_k : \mathcal{D}(k+1) \to \mathcal{D}(k) && \text{for all limit ordinals } k < \lambda
\end{aligned}$$

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 $\mathcal{D}_k : \mathbb{O}_k \to \mathcal{K}$ 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 $i < k$,

$$\mu_{k,i+1} \circ ext_k = F(\mu_{k,i}) : \mathcal{D}(k+1) \to \mathcal{D}(i+1).$$

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 λ-continuous,

$$\mu = \{\mu_i : \mathcal{D}(\lambda) \to \mathcal{D}(i) \mid i < \lambda\}$$

be the limit of $\mathcal{D}$. Then

$$F(\mu) = \{F(\mu_i) : F(\mathcal{D}(\lambda)) \to F(\mathcal{D}(i)) \mid i < \lambda\}$$

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 $ext : \mathcal{D}(\lambda) \to F(\mathcal{D}(\lambda))$—and thus an F-coalgebra—such that for all $i < \lambda$,

$$F(\mu_i) \circ ext = \mu_{i+1} : \mathcal{D}(\lambda) \to \mathcal{D}(i+1).$$

ext is final in $coAlg_F$ and thus, by Lemma 14.1 (2), $\mathcal{D}(\lambda) \cong F(\mathcal{D}(\lambda))$.

Proof. Let $\alpha : A \to F(A)$ 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

$$\nu = \{\nu_i : A \to \mathcal{D}(i) \mid i < \lambda\}$$

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 $unfold^A : A \to \mathcal{D}(\lambda)$ with $\mu_i \circ unfold^A = \nu_i$ for all $i < \lambda$.

Proceed analogously to the proof of Theorem 14.7. ❑

Corollary 14.9

Suppose that all (co)chains of $\mathcal{K}$ have (co)limits. Then the definition of the λ-(co)chain $\mathcal{D}$ in Theorem 14.7 or 14.8 can be extended to the definition of a (co)chain.

If $F : \mathcal{K} \to \mathcal{K}$ is λ-(co)continuous, then $\mathcal{D}$ **converges in λ steps**, i.e., $\mathcal{D}(\lambda) \cong \mathcal{D}(\lambda + 1)$.

Proof. The conjecture follows immediately from Lemma 14.1 and Theorem 14.7 or 14.8.
❏

15 Σ-functors

15.1 Functors for constructive signatures

Let $\Sigma = (S, C)$ be a constructive signature.

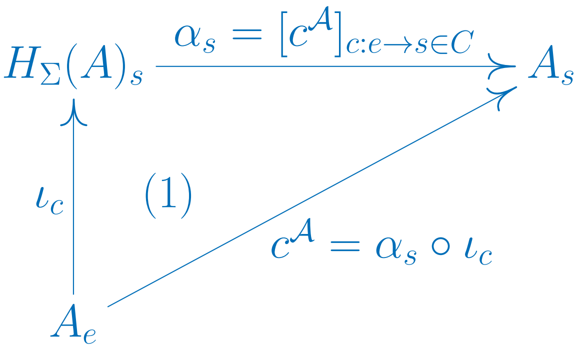
Σ induces the functor $H_\Sigma : Mod(S) \to Mod(S)$:

For all $A, B \in Mod(S)$, $Mod(S)$-morphisms $h : A \to B$ and $s \in S$,

$$\begin{aligned} H_\Sigma(A)_s &=_{def} \textstyle\coprod_{c:e\to s\in C} A_e, \\ H_\Sigma(h)_s &=_{def} \textstyle\coprod_{c:e\to s\in C} h_e. \end{aligned}$$

An H_Σ-algebra $\alpha : H_\Sigma(A) \to A$ (see chapter 14) uniquely corresponds to a Σ-algebra $\mathcal{A}$ with carrier A and vice versa:

For all $s \in S$ and $c : e \to s \in C$,



Hence α_s is the sum extension of the interpretations of all constructors of Σ in A.

Moreover, given Σ-algebras $\mathcal{A}, \mathcal{B}$ and corresponding H_Σ-algebras α, β, an S-sorted function $h : A \to B$ is Σ-homomorphic iff h is an Alg_{H_Σ}-morphism from α to β.

A Σ-algebra $\mathcal{A}$ with carrier A is initial in Alg_Σ iff the corresponding H_Σ-algebra $\alpha : H_\Sigma(A) \to A$ is initial in Alg_{H_Σ}.

Hence by Lemma 14.1 (1), if $\mathcal{A}$ is initial in Alg_Σ, then $[c^\mathcal{A}]_{c:e\to s\in C}$ is iso and thus

- A_s is a sum of $(A_e)_{c:e\to s\in C}$ with injections $c^\mathcal{A} : A_e \to A_s$,
- for all closed $\lambda\Sigma$-term tuples $(t_c : e \to e_s)_{c:e\to s\in C}$, the **case distinction**

  $$case\{c.t_c\}_{c:e\to s\in C} : s \to e_s$$

  has a well-defined interpretation in $\mathcal{A}$:

  $$(case\{c.t_c\}_{c:e\to s\in C})^\mathcal{A} = [t_c^\mathcal{A}]_{c:e\to s\in C} \circ [c^\mathcal{A}]^{-1}_{c:e\to s\in C}$$

  (see chapter 10).

Case distinctions are functional versions of `case`-statements and variant types in the sense of [65] and [1], respectively.

Lemma 15.1 (case distinctions are unique solutions)

$d_s = case\{c.t_c\}^{\mathcal{A}}_{c:e\to s\in C}$ solves

$$\{d_s \circ c^{\mathcal{A}} = t_c^{\mathcal{A}} \mid s \in S\} \tag{1}$$

uniquely in $\mathcal{A}$.

Proof. By Lemma 4.2 (2), for all $c : e \to s \in C$, $\iota_c = [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C} \circ c^{\mathcal{A}}$ and thus

$$case\{c.t_c\}^{\mathcal{A}}_{c:e\to s\in C} \circ c^{\mathcal{A}} = [t_c^{\mathcal{A}}(g)]_{c:e\to s\in C} \circ [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C} \circ c^{\mathcal{A}} = [t_c^{\mathcal{A}}]_{c:e\to s\in C} \circ \iota_c = t_c^{\mathcal{A}}.$$

Hence d_s solves (1) in $\mathcal{A}$.

Conversely, let $d = (d_s : A_s \to A_{e_s})_{s\in S}$ be an S-sorted function such that for all $c : e \to s \in C$, $d_s \circ c^{\mathcal{A}} = t_c^{\mathcal{A}}$ for some closed $\lambda\Sigma$-term t_c. Then by (1),

$$d_s \circ [c^{\mathcal{A}}]_{c:e\to s\in C} \circ \iota_c = d_s \circ c^{\mathcal{A}} = t_c^{\mathcal{A}} = [t_c^{\mathcal{A}}]_{c:e\to s\in C} \circ \iota_c$$

and thus $d_s \circ [c^{\mathcal{A}}]_{c:e\to s\in C} = [t_c^{\mathcal{A}}]_{c:e\to s\in C}$. Hence

$$d_s = d_s \circ [c^{\mathcal{A}}]_{c:e\to s\in C} \circ [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C} = [t_c^{\mathcal{A}}]_{c:e\to s\in C} \circ [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C} = case\{c.t_c\}^{\mathcal{A}}_{c:e\to s\in C},$$

i.e., $case\{c.t_c\}^{\mathcal{A}}_{c:e\to s\in C}$ is the *only* solution of (1) in $\mathcal{A}$. ❑

Given a signature Σ' that includes Σ, we may regard case distinctions as Σ'-formulas (see section 10.1) whose semantics (see section 10.3) is restricted to Σ'-algebras $\mathcal{A}$ such that $\mathcal{A}|_{\Sigma'}$ is initial in Alg_Σ.

Examples

Let A be an S-sorted set and $I, X, Y, Act \subseteq \mathcal{I}$ be as in chapter 8. We omit sort indices if S is a singleton.

$$
\begin{array}{lll}
H_{Mon}(A) & = & 1 + A \times A, \\
H_{Nat}(A) & = & 1 + A, \\
H_{Dyn(X,Y)}(A) & = & X \times A + Y, \\
H_{coStream(X)}(A) & = & X \times A, \\
H_{Bintree(X)}(A) & = & X \times A \times A + 1, \\
H_{Tree(X)}(A) & = & X \times A^*, \\
H_{Reg(X)}(A) & = & A^2 + A^2 + A + \mathcal{P}_+(X) + 2, \\
H_{CCS(Act)}(A) & = & Act + A^2 + A^2 + A \times Act + A \times Act^{Act}.
\end{array}
$$
❏

15.2 Functors for destructive signatures

Let $\Sigma = (S, D)$ be a destructive signature.

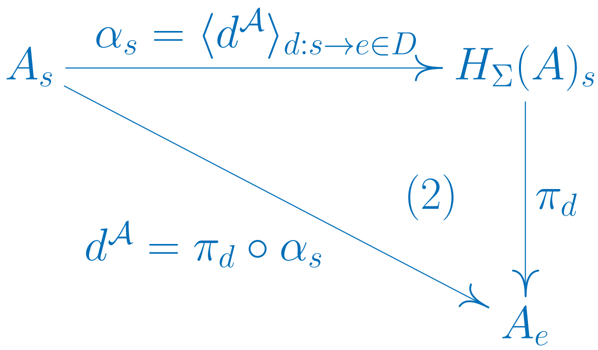
Σ induces the functor $H_\Sigma : Mod(S) \to Mod(S)$:

For all $A, B \in Mod(S)$, $Mod(S)$-morphisms $h : A \to B$ and $s \in S$,

$$H_\Sigma(A)_s =_{def} \prod_{d:s\to e\in D} A_e,$$
$$H_\Sigma(h)_s =_{def} \prod_{d:s\to e\in D} h_e.$$

An H_Σ-coalgebra $\alpha : A \to H_\Sigma(A)$ (see chapter 14) uniquely corresponds to a Σ-algebra $\mathcal{A}$ with carrier A and vice versa:

For all $s \in S$ and $d : s \to e \in D$,



Hence α_s is the product extension of the interpretations of all destructors of Σ in A.

Moreover, given Σ-algebras $\mathcal{A}, \mathcal{B}$ and corresponding H_Σ-coalgebras α, β, an S-sorted function $h : A \to B$ is Σ-homomorphic iff h is a $coAlg_{H_\Sigma}$-morphism from α to β.

A Σ-algebra $\mathcal{A}$ with carrier A is final in Alg_Σ iff the corresponding H_Σ-coalgebra $\alpha : A \to H_\Sigma(A)$ is final in $coAlg_{H_\Sigma}$.

Hence by Lemma 14.1 (2), if $\mathcal{A}$ is final in Alg_Σ, then $\langle d^\mathcal{A} \rangle_{d:s \to e \in D}$ is iso and thus

- A_s is a product of $(A_e)_{d:s \to e \in D}$ with projections $d^\mathcal{A} : A_s \to A_e$,
- for all closed λΣ-term tuples $(t_d : e_s \to e)_{d:s \to e}$, the **object definition**

$$obj\{d.t_d\}_{d:s \to e \in D} : e_s \to s$$

  has a well-defined interpretation in $\mathcal{A}$:

$$obj\{d.t_d\}^\mathcal{A}_{d:s \to e \in D} = \langle d^\mathcal{A} \rangle^{-1}_{d:s \to e \in D} \circ \langle t_d^\mathcal{A} \rangle_{d:s \to e \in D}$$

  (see chapter 10).

Object definitions are functional versions of `merge`-statements and record types in the sense of [65] and [1], respectively.

Lemma 15.2 (object definitions are unique solutions)

$c_s = obj\{d.t_d\}^{\mathcal{A}}_{d:s\to e\in D}$ solves

$$\{d^{\mathcal{A}} \circ c_s = t^{\mathcal{A}}_d \mid s \in S\} \tag{2}$$

uniquely in $\mathcal{A}$.

Proof. By Lemma 4.2 (1), for all $d : s \to e \in D$, $\pi_d = d^{\mathcal{A}} \circ \langle d^{\mathcal{A}}\rangle^{-1}_{d:s\to e\in D}$ and thus

$$d^{\mathcal{A}} \circ obj\{d.t_d\}^{\mathcal{A}}_{d:s\to e\in D} = d^{\mathcal{A}} \circ \langle d^{\mathcal{A}}\rangle^{-1}_{d:s\to e\in D} \circ \langle t^{\mathcal{A}}_d\rangle_{d:s\to e\in D}\circ = \pi_d \circ \langle t^{\mathcal{A}}_d\rangle_{d:s\to e\in D} = t^{\mathcal{A}}_d.$$

Hence c solves (2) in $\mathcal{A}$.

Conversely, let $c = (c_s : A_{e_s} \to A_s)_{s\in S}$ be an S-sorted function such that for all $d : s \to e \in D$, $d^{\mathcal{A}} \circ c_s = t^{\mathcal{A}}_d$ for some closed $\lambda\Sigma$-term t_d. Then by (2),

$$\pi_d \circ \langle d^{\mathcal{A}}\rangle_{d:s\to e\in D} \circ c_s = d^{\mathcal{A}} \circ c_s = t^{\mathcal{A}}_d = \pi_d \circ \langle t^{\mathcal{A}}_d\rangle_{d:s\to e\in D}$$

and thus $\langle d^{\mathcal{A}}\rangle_{d:s\to e\in D} \circ c_s = \langle t^{\mathcal{A}}_d\rangle_{d:s\to e\in D}$. Hence

$$c = \langle d^{\mathcal{A}}\rangle^{-1}_{d:s\to e\in D} \circ \alpha_s \circ c_s = \langle d^{\mathcal{A}}\rangle^{-1}_{d:s\to e\in D} \circ \langle t^{\mathcal{A}}_d\rangle_{d:s\to e\in D} = (obj\{d.t_d\}_{d:s\to e\in D})^{\mathcal{A}},$$

i.e., $obj\{d.t_d\}^{\mathcal{A}}_{d:s\to e\in D}$ is the *only* solution of (2) in $\mathcal{A}$. ❑

Given a signature Σ' that includes Σ, we may regard object definitions as Σ'-formulas (see section 10.1) whose semantics (see section 10.3) is restricted to Σ'-algebras $\mathcal{A}$ such that $\mathcal{A}|_{\Sigma'}$ is final in Alg_Σ.

Examples

Let A be an S-sorted set and $X, Y, Act \subseteq \mathcal{I}$ and $(M, +, 0)$ be a commutative monoid. We omit sort indices if S is a singleton.

$$\begin{array}{lcl}
H_{coNat}(A) & = & A+1, \\
H_{Stream(X)}(A) & = & X \times A, \\
H_{coDyn(X,Y)}(A) & = & X \times A + Y, \\
H_{infBintree(X)}(A) & = & A \times X \times A, \\
H_{coBintree(X)}(A) & = & X \times A \times A + 1, \\
H_{infTree(X)}(A) & = & X \times A^+, \\
H_{coTree_\omega(X)}(A) & = & X \times A^*, \\
H_{coTree(X)}(A)_{tree} & = & X \times A_{trees}, \\
H_{coTree(X)}(A)_{trees} & = & A_{tree} \times A_{trees} + 1,
\end{array}$$

$$
\begin{aligned}
H_{Trans(Act)}(A) &= (Act \times A)^*, \\
H_{WStream(X,M)}(A) &= X \times M_\omega^A, \\
H_{WStream^*(X,M)}(A) &= X \times (A \times M)^*, \\
H_{Med(X)}(A) &= A^X, \\
H_{NMed(X)}(A) &= \mathcal{P}_\omega(A)^X, \\
H_{NMed^*(X)}(A) &= (A^*)^X, \\
H_{WMed(X,M)}(A) &= (M_\omega^A)^X, \\
H_{WMed^*(X,M)}(A) &= ((M \times A)^*)^X, \\
H_{DAut(X,Y)}(A) &= A^X \times Y, \\
H_{Mealy(X,Y)}(A) &= A^X \times Y^X, \\
H_{PAut(X,Y)}(A) &= (1 + A)^X \times Y, \\
H_{NAut(X,Y)}(A) &= \mathcal{P}_\omega(A)^X \times Y, \\
H_{NAut^*(X,Y)}(A) &= (A^*)^X \times Y, \\
H_{WAut(X,M,Y)}(A) &= (M_\omega^A)^X \times Y, \\
H_{WAut^*(X,M,Y)}(A) &= ((A \times M)^*)^X \times Y,
\end{aligned}
$$

$$
\begin{array}{lcl}
H_{PrAut(X,Y)}(A) & = & \mathcal{D}(A)^X \times Y, \\
H_{TAcc(\Sigma)}(A)_s & = & \prod_{c:e \to s \in C} A_e, \quad s \in S, \\
H_{NTAcc(\Sigma)}(A)_s & = & \prod_{c:e \to s \in C} \mathcal{P}_\omega(A_e), \quad s \in S, \\
H_{NTAcc^*(\Sigma)}(A)_s & = & \prod_{c:e \to s \in C} A_e^*, \quad s \in S, \\
H_{Class(BS)}(A) & = & \prod_{i=1}^n ((Y_i \times A) + E_i)^{X_i}, \\
H_{Graph(X,Y)}(A)_{node} & = & X, \\
H_{Graph(X,Y)}(A)_{edge} & = & A_{node} \times A_{node} \times Y. \qquad ❏
\end{array}
$$

Let Σ and Σ′ be both constructive or both destructive signatures with bijective sets of sorts.

Σ and Σ′ are **equivalent** if H_Σ and $H_{\Sigma'}$ are naturally equivalent (modulo renamings of sorts).

Σ′ is a **quotient** of Σ if there is a surjective natural transformation from H_Σ to $H_{\Sigma'}$.

15.3 Final models of destructive non-polynomial signatures

Lemma 15.3 (see [10], 2.4.6/16; [62], 4.3.2/3)

Let $\Sigma = (S, D)$ and $\Sigma' = (S, D')$ be destructive signatures, $\tau : H_\Sigma \rightarrow H_{\Sigma'}$ be a surjective natural transformation, $\mathcal{A}$ be final in Alg_Σ and $\alpha : A \rightarrow H_\Sigma(A)$ be the corresponding H_Σ-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 β there is a $coAlg_{H_{\Sigma'}}$-morphism from β to $\tau_A \circ \alpha$.

Moreover, $\mathcal{A}/{\sim}$ is final in $Alg_{\Sigma'}$ where $\sim$ is the greatest Σ'-bisimulation on A (which, by Theorem 9.6 (2), is a Σ'-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_Σ-coalgebra and thus there is a unique Σ-homomorphism $unfold^B : B \rightarrow A$ from $h \circ \beta$ to α.

$unfold^B$ is also a Σ'-homomorphism from β to $\tau_A \circ \alpha : A \to H_{\Sigma'}(A)$:

$$H_{\Sigma'}(unfold^B) \circ \beta = H_{\Sigma'}(unfold^B) \circ \tau_B \circ h \circ \beta \stackrel{\tau\ natural\ transf.}{=} \tau_A \circ H_\Sigma(unfold^B) \circ h \circ \beta$$
$$\stackrel{unfold^B\ \Sigma-hom.}{=} \tau_A \circ \alpha \circ unfold^B.$$

Hence $nat_\sim \circ unfold^B : B \to A/\sim$ is a Σ'-homomorphism from β to $\tau_A \circ \alpha/\sim$.

It is unique: Let $f, g : B \to A/\sim$ be Σ'-homomorphisms from β to $\tau_A \circ \alpha/\sim$. Then there is an S-sorted function $h : A/\sim \to A$ with $nat_\sim \circ h = id_{A/\sim}$.

Let $\approx$ be the least Σ'-congruence on A that contains all pairs $(h(f(b)), h(f(b)))$ with $b \in B$. Since $\sim$ is the greatest Σ-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

$$\begin{aligned}
\tau_1 : H_{WStream^*(X,M)} &\to H_{WStream(X,M)},\\
\tau_2 : H_{NMed^*(X)} &\to H_{NMed(X)},\\
\tau_3 : H_{WMed^*(X,M)} &\to H_{WMed(X,M)},\\
\tau_4 : H_{NAut^*(X,Y)} &\to H_{NAut(X,Y)},\\
\tau_5 : H_{WAut^*(X,M,Y)} &\to H_{WAut(X,M,Y)},\\
\tau_6 : H_{WAut^*(X,\mathbb{R}_{\geq 0},Y)} &\to H_{PrAut(X,Y)},\\
\tau_7 : H_{NTAcc^*(\Sigma)} &\to H_{NTAcc(\Sigma)}
\end{aligned}$$

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 $(as_c)_{c:e\to s\in C} \in \prod_{c:e\to s\in C} A_e^* = H_{NTAcc^*(\Sigma)}(A)$,

$$\tau_{7,A}((as_c)_{c:e\to s\in C}) = (\{\pi_i(as_c) \mid 1 \leq i \leq |as_c|\})_{c:e\to s\in C}$$
$$\in \prod_{c:e\to s\in C} \mathcal{P}_\omega(A_e) = H_{NTAcc(\Sigma)}(A).$$

$\tau_1, \ldots, \tau_7$ are surjective natural transformations.

Hence by Lemma 15.3, 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 Σ such that a final Σ'-algebra is given by a quotient of the final Σ-algebra.

Let us take a closer look at τ_2 (see above),

$$\Sigma = NMed^*(X) = (\{state\}, \{\delta' : state \to (state^*)^X\})$$

and

$$\Sigma' = NMed(X)[\delta'/\delta] = (\{state\}, \{\delta' : state \to \mathcal{P}_\omega(state)^X\})$$

(see section 8.3). $\mathcal{A} = NPow^*(X)$ with carrier $T = otr(X \times \mathbb{N}, 1)$ is final in Alg_Σ (see sample final algebra 9.18.19). Hence by Lemma 15.3, $\mathcal{A}$ is weakly final in $Alg_{\Sigma'}$ for $\delta'^{\mathcal{A}} =_{def} \pi_{\delta'} \circ \tau_{2,T} \circ \delta^{\mathcal{A}}$, i.e., for all $\{n_x \mid x \in X\} \subseteq \mathbb{N}$, $t = ()\{(x,i) \to t_{x,i} \mid x \in X,\ 1 \le i \le n_x\} \in T$ and $x \in X$,

$$\delta'^{\mathcal{A}}(t)(x) = \{t_{x,1}, \dots, t_{x,n_x}\} \quad \text{if} \quad \delta^{\mathcal{A}}(t)(x) = (t_{x,1}, \dots, t_{x,n_x})$$

(see sample algebra 9.6.32).

Lemma 15.3 also implies that the quotient $\mathcal{A}/{\sim}$ with carrier $T/{\sim}$ is final in $Alg_{\Sigma'}$ where $\sim$ is the greatest binary relation on T such that for all $t, u \in T$ and $x \in X$,

$$t \sim u \quad \text{implies} \quad \delta'^{\mathcal{A}}(t)(x) \sim_{\mathcal{P}_\omega(X)} \delta'^{\mathcal{A}}(u)(x),$$

i.e., for all $\{(m_x, n_x) \mid x \in X\} \subseteq \mathbb{N}^2$, $t = ()\{(x,i) \to t_{x,i} \mid x \in X,\ 1 \leq i \leq m_x\}$, $u = ()\{(x,i) \to u_{x,i} \mid x \in X,\ 1 \leq i \leq n_x\} \in T$ and $x \in X$,

$$t \sim u \quad \text{implies} \quad \{t_{x,1}, \dots, t_{x,m_x}\} \sim_{\mathcal{P}_\omega(X)} \{u_{x,1}, \dots, u_{x,n_x}\}$$

and thus

$$t \sim u \quad \text{implies} \quad \begin{cases} \forall\, i \in [m_x]\ \exists\, j \in [n_x] : t_{x,i} \sim u_{x,j}, \\ \forall\, i \in [n_x]\ \exists\, j \in [m_x] : u_{x,i} \sim t_{x,j}. \end{cases}$$

15.4 From constructors to destructors

Let $\Sigma = (S, C)$ be a constructive polynomial signature, $C_s = \{c \in C \mid trg(c) = s\}$,

$$\begin{aligned} D &= \{d_s : s \to \coprod_{c:e\to s\in C_s} e \mid s \in S\}, \\ co\Sigma &= (S, D), \end{aligned}$$

$\mathcal{A}$ be an initial Σ-algebra with carrier A and $B = \bigcup \mathcal{I}$.

By Lemma 14.1 (1), the initial H_Σ-algebra

$$\alpha = \{\alpha_s : H_\Sigma(A)_s \stackrel{[c^{\mathcal{A}}]_{c:e\to s\in C}}{\longrightarrow} A_s \mid s \in S\}$$

is iso (see chapter 15). Consequently,

$$\{\alpha_s^{-1} : A_s \to H_\Sigma(A)_s \mid s \in S\}$$

is an H_Σ-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 $c : e \to s \in C$,

$$d_s^{\mathcal{B}} \circ c^{\mathcal{A}} = \alpha_s^{-1} \circ c^{\mathcal{A}} \stackrel{(1)}{=} \alpha_s^{-1} \circ [c^{\mathcal{A}}]_{c:e\to s\in C} \circ \iota_c = \alpha_s^{-1} \circ \alpha_s \circ \iota_c = \iota_c.$$

Since $co\Sigma$ is destructive, Theorem 9.12 implies that $DT_{co\Sigma}$ is final in $Alg_{co\Sigma}$.

CT_Σ and, analogously, T_Σ are $co\Sigma$-algebras:

For all $c : e \to s \in C$ and $t \in CT_{\Sigma,e}$,

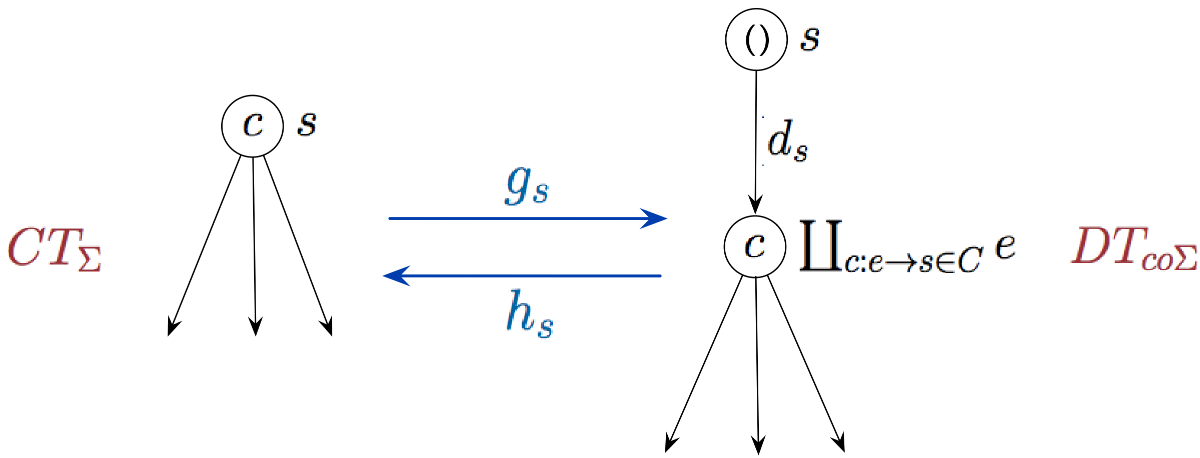
$$d_s^{CT_\Sigma}(c(t)) \ =_{def} \ c(t) \in \coprod_{c:e\to s\in C} CT_{\Sigma,e}.$$

$DT_{co\Sigma}$ and, analogously, $coT_{co\Sigma}$ are Σ-algebras:

For all $c : e \to s \in C$ and $t \in DT_{co\Sigma,e}$,

$$c^{DT_{co\Sigma}}(t) \ =_{def} \ ()\{d_s \to c(t)\} \in DT_{co\Sigma,s}.$$

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 $g : CT_\Sigma \to DT_{co\Sigma}$ and $h : DT_{co\Sigma} \to CT_\Sigma$ are defined inductively on $(D \cup B)^*$ as follows:

For all $c : e \to s \in C$, $t \in CT_{\Sigma,e}$ and $t' \in DT_{co\Sigma,e}$,

$$\begin{aligned} g_s(c(t)) &= ()\{d_s \to c(g_e(t))\}, \\ h_s(()\{d_s \to c(t')\}) &= c(h_e(t')). \end{aligned}$$

Bijective S-sorted functions $g : T_\Sigma \to coT_{co\Sigma}$ and $h : coT_{co\Sigma} \to T_\Sigma$ 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 Σ-homomorphic:

For all $c : e \to s \in C$, $t \in CT_{\Sigma,e}$ and $t' \in DT_{co\Sigma,e}$,

$$g_{\coprod_{c:e\to s\in C} e}(d_s^{CT_\Sigma}(c(t))) = g_{\coprod_{c:e\to s\in C} e}(c(t)) = g_{\coprod_{c:e\to s\in C} e}(\iota_c(t)) = \iota_c(g_e(t)) = c(g_e(t))$$

$$= d_s^{DT_{co\Sigma}}(()\{d_s \to c(g_e(t))\}) = d_s^{DT_{co\Sigma}}(g_s(c(t))),$$

$$h_s(c^{DT_{co\Sigma}}(t')) = h_s(()\{d_s \to c(t')\}) = c(h_e(t')) = c^{CT_\Sigma}(h_e(t')).$$

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 9.1 (1), h is $co\Sigma$-homomorphic.

Since h is Σ-homomorphic and $g \circ h = id$, $g \circ h$ and thus h are mono in Alg_Σ. Hence by Lemma 9.1 (2), g is Σ-homomorphic. ❑

Therefore, CT_Σ and $DT_{co\Sigma}$ and, analogously, T_Σ and $coT_{co\Sigma}$ are both Σ- and $co\Sigma$-isomorphic. Consequently, CT_Σ **is final in** $Alg_{co\Sigma}$ and $coT_{co\Sigma}$ **is initial in** Alg_Σ.

Given a $co\Sigma$-algebra $\mathcal{A}$ with carrier A, the above definition of the bijection $h : DT_{co\Sigma} \to CT_\Sigma$ implies that the S-components of $unfold'^{\mathcal{A}} =_{def} h \circ unfold^{\mathcal{A}} : A \to CT_\Sigma$ (see section 9.16) are defined as follows: For all $c : e \to s \in C$, $a \in A_s$ and $b \in A_e$,

$$d_s^{\mathcal{A}}(a) = \iota_c(b) \;\text{ implies }\; unfold_s'^{\mathcal{A}}(a) = c(unfold_e'^{\mathcal{A}}(b)).$$

Proof. Let $d_s^{\mathcal{A}}(a) = \iota_c(b)$. Then

$$unfold^{\mathcal{A}}(a) = ()\{d_s \to unfold^{\mathcal{A}}(\iota_c(b))\} = ()\{d_s \to c(unfold^{\mathcal{A}}(b))\}. \tag{2}$$

Hence

$$unfold'^{\mathcal{A}}(a) = h(unfold^{\mathcal{A}}(a)) \overset{(1)}{=} h(()\{d_s \to c(unfold^{\mathcal{A}}(b))\}) = c(h(unfold^{\mathcal{A}}(b)))$$
$$= c(unfold'^{\mathcal{A}}(b)).$$
❑

15.5 From destructors to constructors

Let $\Sigma = (S, D)$ be a destructive polynomial signature, $D_s = \{d \in D \mid src(d) = s\}$,

$$\begin{aligned} C &= \{c_s : \textstyle\prod_{d:s\to e \in D_s} e \to s \mid d \in C_s,\ s \in S\}, \\ co\Sigma &= (S, C), \end{aligned}$$

$\mathcal{A}$ be a final Σ-algebra with carrier A and $B = \bigcup \mathcal{I}$.

By Lemma 14.1 (2), the final H_Σ-coalgebra

$$\alpha = \{\alpha_s : A_s \stackrel{\langle d^{\mathcal{A}} \rangle_{d:s\to e\in D}}{\longrightarrow} H_\Sigma(A)_s \mid s \in S\}$$

is iso (see chapter 15). Consequently,

$$\{\alpha_s^{-1} : H_\Sigma(A)_s \to A_s \mid s \in S\}$$

is an H_Σ-algebra, which corresponds to the $co\Sigma$-algebra $\mathcal{B}$ that is defined as follows:

For all $s \in S$, $B(s) = A_s$ and $c_s^{\mathcal{B}} = \alpha_s^{-1}$. Hence for all $d : s \to e \in D$,

$$d^{\mathcal{A}} \circ c_s^{\mathcal{B}} = d^{\mathcal{A}} \circ \alpha_s^{-1} \stackrel{(2)}{=} \pi_d \circ \langle d^{\mathcal{A}} \rangle_{d:s\to e\in D} \circ \alpha_s^{-1} = \pi_d \circ \alpha_s \circ \alpha_s^{-1} = \pi_d.$$

Since $co\Sigma$ is constructive, Theorem 9.7 implies that $T_{co\Sigma}$ is initial in $Alg_{co\Sigma}$.

DT_Σ and, analogously, coT_Σ are $co\Sigma$-algebras:

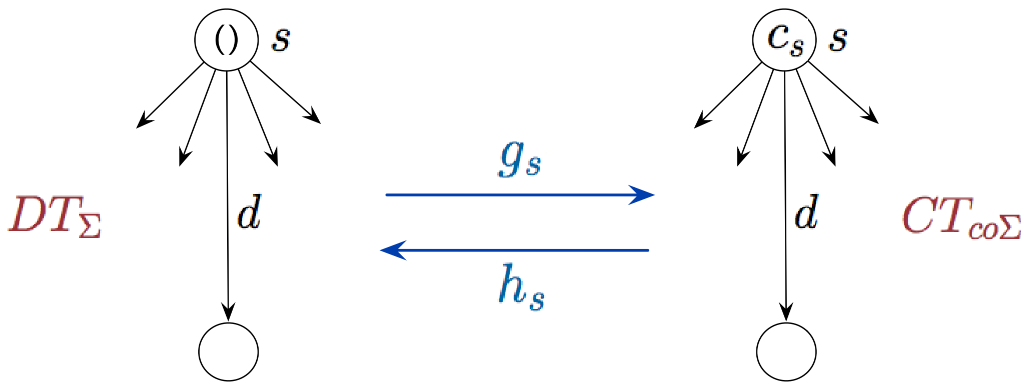
For all $s \in S$ and $t = (t_d)_{d:s\to e\in D} \in \prod_{d:s\to e\in D} DT_{\Sigma,e}$,

$$c_s^{DT_\Sigma}(t) \ =_{def} \ (\,)\{d \to t_d \mid d : s \to e \in D\} \in DT_{\Sigma,s}.$$

$CT_{co\Sigma}$ and, analogously, $T_{co\Sigma}$ are Σ-algebras:

For all $d : s \to e \in D$ and $(t_d)_{d:s\to e\in D} \in \prod_{d:s\to e\in D} CT_{co\Sigma,e}$,

$$d^{CT_{co\Sigma}}((\,)\{d \to t_d \mid d : s \to e \in D\}) \ =_{def} \ t_d \in CT_{co\Sigma,e}.$$



The values of S-sorted functions $g : DT_\Sigma \to CT_{co\Sigma}$ and $h : CT_{co\Sigma} \to DT_\Sigma$ are defined inductively on $(D \cup B)^*$ as follows:

For all $s \in S$, $t = ()\{d \to t_d \mid d : s \to e \in D\} \in DT_{\Sigma,s}$ and $t' = c_s\{d \to t_d\} \mid d : s \to e \in D\} \in CT_{co\Sigma,s}$,

$$\begin{aligned} g_s(t) &= c_s\{d \to g_e(t_d) \mid d : s \to e \in D\}, \\ h_s(t') &= ()\{d \to h_e(t_d) \mid d : s \to e \in D\}. \end{aligned}$$

Bijective S-sorted functions $g : coT_\Sigma \to T_{co\Sigma}$ and $h : T_{co\Sigma} \to coT_\Sigma$ 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 Σ-homomorphic:

For all $s \in S$, $t = ()\{d \to t_d \mid d : s \to e \in D\} \in DT_{\Sigma,s}$ and $t' = c_s\{d \to t_d\} \mid d : s \to e \in D\} \in CT_{co\Sigma,s}$,

$$\begin{aligned} &g_s(c_s^{DT_\Sigma}(t)) = g_s(()\{d \to t_d \mid d : s \to e \in D\}) \\ &= c_s\{d \to g_e(t_d) \mid d : s \to e \in D\} = c_s^{CT_{co\Sigma}}\{d \to g_e(t_d) \mid d : s \to e \in D\} \end{aligned}$$

$$= c_s^{CT_{co\Sigma}}(g_{\prod_{d:s\to e\in D}}(()\{d\to t_d \mid d:s\to e\in D\})) = c_s^{CT_{co\Sigma}}(g_{\prod_{d:s\to e\in D}}(t)),$$
$$h_e(d^{CT_{co\Sigma}}(t')) = h_e(t_d) = d^{DT_\Sigma}(()\{d\to h_e(t_d) \mid d:s\to e\in D\}) = d^{DT_\Sigma}(h_s(t')).$$

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 9.1 (1), h is $co\Sigma$-homomorphic.

Since h is Σ-homomorphic and $g\circ h = id$, $g\circ h$ and thus h are mono in Alg_Σ. Hence by Lemma 9.1 (2), g is Σ-homomorphic. ❑

Therefore, DT_Σ and $CT_{co\Sigma}$ and, analogously, coT_Σ and $T_{co\Sigma}$ are both Σ- and $co\Sigma$-isomorphic. Consequently, $CT_{co\Sigma}$ **is final in** Alg_Σ and coT_Σ **is initial in** $Alg_{co\Sigma}$.

Given a Σ-algebra $\mathcal{A}$ with carrier A, the above definition of the bijection $g: coT_\Sigma \to T_{co\Sigma}$ implies that the S-components of $unfold'^{\mathcal{A}} = g\circ unfold^{\mathcal{A}} : A\to CT_{co\Sigma}$ (see section 9.16) are defined as follows: For all $s\in S$ and $a\in A_s$,

$$unfold_s'^{\mathcal{A}}(a) = c_s\{d\to unfold_e'^{\mathcal{A}}(d^{\mathcal{A}}(a)) \mid d:s\to e\in D\}.$$

Proof. $unfold'^{\mathcal{A}}(a) = g(unfold^{\mathcal{A}}(a)) = g(()\{d\to unfold^{\mathcal{A}}(d^{\mathcal{A}}(a)) \mid d:s\to e\in D\})$
$$= c_s\{d\to g(unfold^{\mathcal{A}}(d^{\mathcal{A}}(a))) \mid d:s\to e\in D\}$$
$$= c_s\{d\to unfold'^{\mathcal{A}}(d^{\mathcal{A}}(a)) \mid d:s\to e\in D\}.$$ ❑

15.6 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 ω-CPOs as objects and strict and ω-continuous functions as morphisms (see section 3).

$Poset^S$ 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 $A \in Set^{\mathcal{T}_{po}(S)}$ is lifted to an object of $Poset^{\mathcal{T}_{po}(S)}$ and $CPO^{\mathcal{T}_{po}(S)}$ as follows:

- $A_1 = 1$.
- For all $I \in \mathcal{I}$ and $(e_i)_{i \in I} \in \mathcal{T}_{po}(S)^I$,

$$A_{\prod_{i \in I} e_i} = \prod_{i \in I} A_{e_i} \quad \text{and} \quad A_{\coprod_{i \in I} e_i} = \coprod_{i \in I} A_{e_i} \cup \{\perp_{\coprod_{i \in I} e_i}\}.$$

The partial orders, least elements and suprema of ω-chains are defined as follows:

- For all $I \in \mathcal{I}$, $(e_i)_{i\in I} \in \mathcal{T}_{po}(S)^I$, $a, b \in A_{\prod_{i\in I} e_i}$ and ω-chains C of $A_{\prod_{i\in I} e_i}$ and $i \in I$,

$$\begin{array}{rcl} a \leq^A_{\prod_{i\in I} e_i} b & \Leftrightarrow & \forall\ i \in \mathbb{N} : \pi_i(a) \leq_{e_i} \pi_i(b), \\ \pi_i(\bot^A_{\prod_{i\in I} e_i}) & = & \bot^A_{e_i}, \\ \pi_i(\bigsqcup C) & = & \bigsqcup\{\pi_i(a) \mid a \in C\}. \end{array}$$

- For all $I \in \mathcal{I}$, $(e_i)_{i\in I} \in \mathcal{T}_{po}(S)^I$, $a, b \in A_{\coprod_{i\in I} e_i}$ and ω-chains C of $A_{\coprod_{i\in I} e_i}$,

$$\begin{array}{rcl} a \leq^A_{\coprod_{i\in I} e_i} b & \Leftrightarrow & a = \bot_{\coprod_{i\in I} e_i} \vee \\ & & \exists\ i \in I,\ a', b' \in A_{e_i} : a' \leq^A_{e_i} b' \wedge \iota_i(a') = a \wedge \iota_i(b') = b. \\ \bigsqcup C & = & \begin{cases} \bot_{\coprod_{i\in I} e_i} & \text{if } \forall\ C = \{\bot_{\coprod_{i\in I} e_i}\}, \\ \bigsqcup\{a \in A_{e_i} \mid \iota_i(a) \in C,\ i \in I\} & \text{otherwise.} \end{cases} \end{array}$$

Let $\Sigma = (S, F)$ be a signature.

$PAlg_\Sigma$ denotes the category of all Σ-algebras with carrier $A \in Poset^S$, monotonic operations (w.r.t. the above lifting of A to an object of $Poset^{\mathcal{T}_{po}(S)}$) and all Σ-homomorphisms in $Mor(Poset^S)$. The objects of $PAlg_\Sigma$ are called **monotone Σ-algebras**.

$CAlg_\Sigma$ denotes the category of all Σ-algebras with carrier $A \in CPO^S$ and ω-continuous operations (w.r.t. the above lifting of A to an object of $CPO^{\mathcal{T}_{po}(S)}$) and all Σ-homomorphisms in $Mor(CPO^S)$. The objects of $CAlg_\Sigma$ are called **ω-continuous Σ-algebras**.

Proposition 15.4

Let $\mathcal{A}$ be a monotone Σ-algebra with carrier A such that for all $s \in S$, A_s is chain-finite (see chapter 3). Then $\mathcal{A}$ is ω-continuous.

Proof. Since for all $e \in \mathcal{T}_{po}(S)$, A_e is chain-finite, Proposition 3.3 (4) implies that all operations of $\mathcal{A}$ are ω-continuous. ❏

Both terms and flowcharts form a CPO

Let $\Sigma = (S, C)$ be a constructive polynomial signature and V, C be as in section 9.3.

The sets $CT_\Sigma^\perp(V)$ and $T_\Sigma^\perp(V)$ of (well-founded) **ordered Σ-terms over** V are defined the same as $CT_\Sigma(V)$ and $T_\Sigma(V)$, respectively, except that (1), (2), (3) and (5) in section 9.3 are replaced as follows:

- For all $s \in S$ and $t \in M_s$, $t \in V_s \cup \{\Omega\}$ (see chapter 2) or there are $c : e \to s \in C$ and $u \in M_e$ such that $t = c(u)$. (1')
- For all $e = \coprod_{i \in I} \prod_{j \in J} e_{ij} \in \mathcal{T}_s(S)$ and $t \in M_e$, $t = \Omega$ or there are $i \in I$ and $u \in \times_{j \in J} M_{e_{ij}}$ such that $t = i(u)$. (2')
- For all $s \in S$, $V_s \cup \{\Omega\} \subseteq M_s$. (3')
- For all $e = \coprod_{i \in I} \prod_{j \in J} e_{ij} \in \mathcal{T}_s(S)$, $i \in I$ and $t \in \times_{j \in J} M_{e_{ij}}$, $\Omega, i(t) \in M_e$. (5')

Let $V \in Set_b^S$ (see chapter 7) and for all $s \in S$, let $V_s = \emptyset$. Then the elements of $CT_\Sigma^\perp =_{def} CT_\Sigma^\perp(V)$ und $T_\Sigma^\perp =_{def} T_\Sigma^\perp(V)$ are called **ground ordered Σ-terms**.

Let $\Sigma = (S, D)$ be a destructive polynomial signature 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 Σ-flowcharts over** V are defined the same as $\overline{CT_\Sigma}(V)$ and $\overline{T_\Sigma}(V)$, respectively, except that (1), (2), (3) and (5) in section 9.19 are replaced as follows:

- For all $s \in S$ and $t \in M_s$, $t \in V_s \cup \{\Omega\}$ (see chapter 2) or there are $d : s \to e \in F$ and $u \in M_e$ such that $t = d(u)$. (1')
- For all $e = \prod_{i\in I}\coprod_{j\in J} e_{ij} \in \mathcal{T}_p(S)$ and $t \in M_e$, $t = \Omega$ or there are $i \in I$ and $u \in \times_{j\in J} M_{e_{ij}}$ such that $t = i(u)$. (2')
- For all $s \in S$, $V_s \cup \{\Omega\} \subseteq M_s$. (3')
- For all $e = \prod_{i\in I}\coprod_{j\in J} e_{ij} \in \mathcal{T}_s(S)$, $i \in I$ and $t \in \times_{j\in J} M_{e_{ij}}$, $\Omega, i(t) \in M_e$. (5')

$T_\Sigma^\perp(V), \overline{T_\Sigma^\perp}(V) \in Poset^S$ and $CT_\Sigma^\perp(V), \overline{CT_\Sigma^\perp}(V) \in CPO^S$.

Proof.

For all $s \in S$, Ω 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 $t, t' \in T$,

$$t \leq_s t' \quad \Leftrightarrow_{def} \quad \forall\, w \in def(t) : t(w) = t'(w).$$

Every ω-chain $T \subseteq CT_\Sigma^\perp(V)_s$ or $T \subseteq \overline{CT_\Sigma^\perp}(V)_s$ has a supremum: For all $w \in B^*$,

$$(\bigsqcup T)(w) = \begin{cases} t(w) & \text{if } \exists\, t \in T : w \in def(t), \\ \perp & \text{otherwise.} \end{cases} \quad ❑$$

$T_\Sigma^\perp(V) \in PAlg_\Sigma$ and $CT_\Sigma^\perp(V) \in CAlg_\Sigma$.

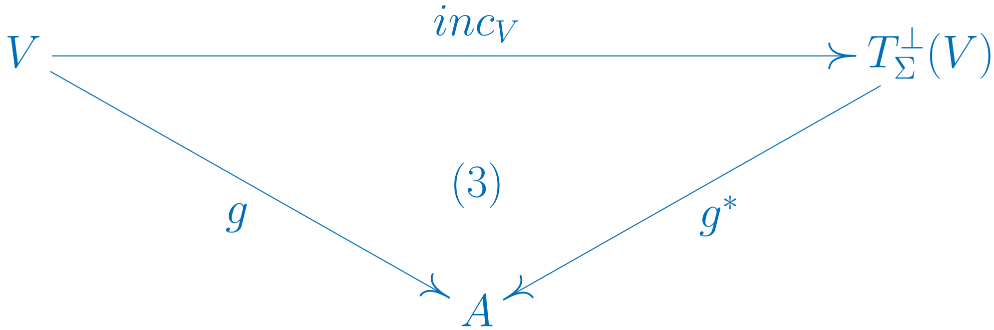
Proof. The arrows of Σ are interpreted in $CT_\Sigma(V)^\perp$ as in $CT_\Sigma(V)$. The interpretations are ω-continuous.

$T_\Sigma^\perp(V)$ is a monotone Σ-subalgebra of $CT_\Sigma^\perp(V)$. ❑

Theorem 15.5 (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 Σ-algebra with carrier A and $g \in A^V$.



The **monotone term extension** $g^* : T_\Sigma^\perp(V) \to A$ of g is the S-sorted function that is defined on $T_\Sigma(V)$ the same as $g^* : T_\Sigma(V) \to A$ (see section 9.11). In addition,

- for all $s \in S$, $g^*_s(\Omega) = \perp^A_s$.

g^* is strict, monotone and Σ-homomorphic.

The uniqueness of g^* w.r.t. (3) can be shown analogously to the proof of Theorem 9.7. Hence we conclude that $T_\Sigma^\perp(V)$ is free over V in $PAlg_\Sigma$. ❏

Let $\Sigma = (S, C)$ be a constructive polynomial signature. For all $n \in \mathbb{N}$, the $\mathcal{T}_s(S)$-sorted function $\_|_n : CT^{\perp}_{\Sigma}(V) \to T^{\perp}_{\Sigma}(V)$ is defined inductively as follows:

For all $t \in CT^{\perp}_{\Sigma}(V)$, $x \in V \cup \{\Omega\}$, $c : \prod_{i \in I} e_i \to s \in C$, $u = (u_i)_{i \in I} \in \times_{i \in I} T^{\perp}_{\Sigma}(V)_{e_i}$, $I \in \mathcal{I}$, $i \in I$ and $t_i \in T^{\perp}_{\Sigma}(V)_{e_i}$,

$$\begin{aligned} t|_0 &= \Omega, \\ x|_{n+1} &= x, \\ c(u)|_{n+1} &= c(u_i|_n)_{i \in I}, \\ i(t_i)|_{n+1} &= i(t_i|_n). \end{aligned}$$

Hence $t = \bigsqcup_{n<\omega} t|_n$.

Given a Σ-algebra $\mathcal{A}$ with carrier A, we extend the functional interpretation of well-founded Σ-terms to arbitrary ones: For all $t \in CT^{\perp}_{\Sigma}(V)$,

$$t^{\mathcal{A}} =_{def} \bigsqcup_{n<\omega} (t|_n)^{\mathcal{A}}.$$

Let $\Sigma = (S, D)$ be a destructive polynomial signature. For all $n \in \mathbb{N}$, the $\mathcal{T}_p(S)$-sorted function $\_|_n : \overline{CT_\Sigma^\perp}(V) \to \overline{T_\Sigma}^\perp(V)$ is defined inductively as follows:

For all $t \in \overline{CT_\Sigma^\perp}(V)$, $x \in V \cup \{\Omega\}$, $d : s \to \coprod_{i\in I} e_i \in D$, $u = (u_i)_{i\in I} \in \times_{i\in I} T_\Sigma^\perp(V)_{e_i}$, $I \in \mathcal{I}$, $i \in I$ and $t_i \in \overline{T_\Sigma}^\perp(V)_{e_i}$,

$$\begin{aligned} t|_0 &= \Omega, \\ x|_{n+1} &= x, \\ d(u)|_{n+1} &= d(u_i|_n)_{i\in I}, \\ i(t_i)|_{n+1} &= i(t_i|_n). \end{aligned}$$

Hence $t = \bigsqcup_{n<\omega} t|_n$.

Given a Σ-algebra $\mathcal{A}$ with carrier A, we extend the interpretation of well-founded Σ-flowcharts to arbitrary ones: For all $t \in \overline{CT_\Sigma^\perp}(V)$,

$$t^{\mathcal{A}} =_{def} \bigsqcup_{n<\omega} (t|_n)^{\mathcal{A}}.$$

Theorem 15.6 (ω-Completion Theorem)

Let $A \in CPO^S$, Σ be a constructive polynomial signature and $f : T_\Sigma^\perp(V) \to A$ be strict and monotone. Then

$$\begin{aligned} f_\omega : CT_\Sigma^\perp(V) &\to A \\ t &\mapsto \bigsqcup\{f(t|_n) \mid n \in \mathbb{N}\} \end{aligned}$$

is strict and ω-continuous.

Moreover, if A is the carrier of an ω-continuous Σ-algebra and f is Σ-homomorphic, then f_ω is Σ-homomorphic.

Let $A \in CPO^S$, Σ be a destructive polynomial signature and $f : \overline{T_\Sigma}^\perp(V) \to A$ be strict and monotone. Then

$$\begin{aligned} f_\omega : \overline{CT_\Sigma^\perp}(V) &\to A \\ t &\mapsto \bigsqcup\{f(t|_n) \mid n \in \mathbb{N}\} \end{aligned}$$

is strict and ω-continuous.

Proof. See the proof of [55], Thm. 4.8. ❏

Theorem 15.7 (generalization of [55], Cor. 4.9)

Let $\Sigma = (S, C)$ 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.

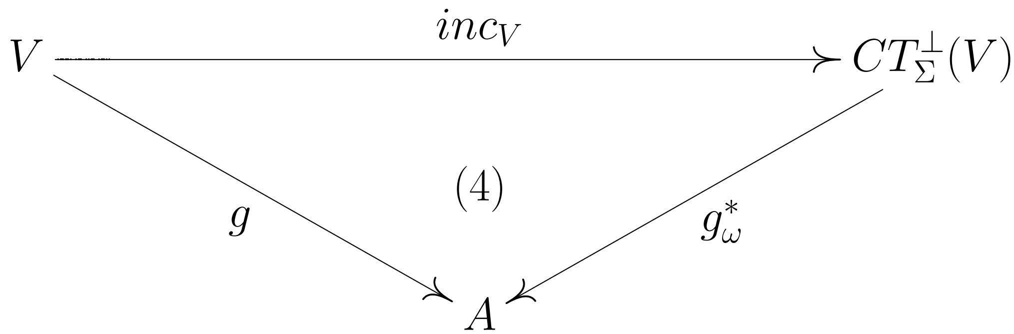
For all $c : e \to s \in C$, $c^{CT_\Sigma^\perp(V)}$ is ω-continuous:

Let T be an ω-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 ω-continuous Σ-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 Σ-homomorphism $g^* : T_\Sigma^\perp(V) \to A$.

By Theorem 15.6, $g^*_\omega : CT_\Sigma^\perp(V) \to A$ is strict, ω-continuous and Σ-homomorphic.



For the proof that there is at most one strict and ω-continuous Σ-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^*_ω no longer depends on g and thus agrees with the ω-completion $fold^{\mathcal{A}}_\omega : CT_\Sigma^\perp \to A$ of the unique monotonic Σ-homomorphism $fold^{\mathcal{A}} : T_\Sigma^\perp \to A$ (see Theorem 15.5). ❑

We conclude that non-well-founded elements of CT_Σ can be regarded as suprema of ω-chains of well-founded ones. Together with the initiality of T_Σ in Alg_Σ and the finality of CT_Σ in $Alg_{co\Sigma}$, Theorem 15.7 entails the following corollary:

The final $co\Sigma$-algebra is a completion of the initial Σ-algebra (see [21], Thm. 3.2; [5], Prop. IV.2).

Lemma 15.8 (Substitutionslemma)

Let $\Sigma = (S, C)$ be a constructive polynomial signature and V, V' be S-sorted sets of variables. For all Σ-algebras $\mathcal{A}$ with carrier A, substitutions $g : V \to CT_\Sigma^\perp(V')$ and term valuations $h : V' \to A$,

$$(h^*_\omega \circ g)^* = h^*_\omega \circ g^* : T_\Sigma(V) \to CT_\Sigma^\perp(V'). \quad (1)$$

Proof. By Theorem 15.6, h^*_ω is Σ-homomorphic.

Hence by Lemma 9.9, (1) holds true. ❑

Lemma 15.9 (Substitutionslemma) ****

Let $\Sigma = (S, D)$ be a destructive polynomial signature and V, V' be S-sorted sets of variables. For all Σ-algebras $\mathcal{A}$ with carrier A, flowchart substitutions $g : V \to \overline{CT^{\perp}_{\Sigma}}(V')$ and flowchart valuations $h : V' \to B^A$,

$$(h^+_\omega \circ g)^+ = h^+_\omega \circ g^* : (\overline{T_\Sigma}(V)_e \to B^{A_e})_{e \in \mathcal{T}_{po}(S)}.$$

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 $d : s \to e \in D$ and $u \in \overline{T_\Sigma}(V)$. Then

$$(h^+_\omega \circ g)^+(t) = (h^+_\omega \circ g)^+(u) \circ d^{\mathcal{A}} \overset{ind.\ hyp.}{=} h^+_\omega(g^*(u)) \circ d^{\mathcal{A}} = (\textstyle\bigsqcup_{n<\omega} h^+(g^*(u)|_n)) \circ d^{\mathcal{A}}$$
$$= \textstyle\bigsqcup_{n<\omega}(h^+(g^*(u)|_n) \circ d^{\mathcal{A}}) = \bigsqcup_{n<\omega} h^+(d(g^*(u)|_n)) = \bigsqcup_{n<\omega} h^+(d(g^*(u))|_{n+1})$$

$$= \bigsqcup_{n<\omega} h^+(d(g^*(u))|_n) = h^+_\omega(d(g^*(u))) = h^+_\omega(g^*(d(u))) = h^+_\omega(g^*(t)).$$

Case 3. $t = i(u)$ for some $i \in I$, $I \in \mathcal{I}$, $u \in \overline{T_\Sigma}(V)_e$ and $e = \prod_{i\in I} e_i \in \mathcal{T}_{po}(S)$. Then (2) follows analogously to Case 2 with i instead of d and π_i instead of $d^{\mathcal{A}}$.

Case 4. $t = ()\{i \to t_i \mid i \in I\}$ for some $I \in \mathcal{I}$, $(t_i)_{i\in I} \in \mathsf{X}_{i\in I}\overline{T_\Sigma}(V)_{e_i}$ and $e = \coprod_{i\in I} e_i \in \mathcal{T}_{po}(S)$. Then

$$(h^+_\omega \circ g)^+(t) = (h^+_\omega \circ g)^+(()\{i \to t_i \mid i \in I\}) = [(h^+_\omega \circ g)^+(t_i)]_{i\in I} \stackrel{ind.\ hyp.}{=} [h^+_\omega(g^*(t_i))]_{i\in I}$$
$$= [\bigsqcup_{n<\omega} h^+(g^*(t_i)|_n)]_{i\in I} = \bigsqcup_{n<\omega}[h^+(g^*(t_i)|_n)]_{i\in I} = \bigsqcup_{n<\omega} h^+(()\{i \to g^*(t_i)|_n \mid i \in I\})$$
$$= \bigsqcup_{n<\omega} h^+(()\{i \to g^*(t_i) \mid i \in I\}|_{n+1}) = \bigsqcup_{n<\omega} h^+(g^*(t)|_{n+1}) = \bigsqcup_{n<\omega} h^+(g^*(t)|_n)$$
$$= h^+_\omega(g^*(t)).$$ ❑

16.1 Three criteria

Let $\Sigma = (S, F)$ be a signature, V be a $\mathcal{T}(S)$-sorted set of variables and $\mathcal{A}$ be a Σ-algebra with carrier A.

Theorem 16.1

Let $C\Sigma = (S, C)$ be a constructive polynomial subsignature of Σ, $\mathcal{A}|_{C\Sigma}$ be initial in $Alg_{C\Sigma}$ and $D = \{d_s : s \to e_s \mid s \in S\}$ be a set of polynomial destructors.

For all $c : e \to s \in C$, let $\overline{c} : e[e_s/s \mid s \in S] \to e_s$ be a closed $\lambda\Sigma$-term.

There is a unique solution $g \in A^C$in $\mathcal{A}$ of the Σ-formulas

$$\bigwedge_{c:e\to s\in C} d_s \circ c = \overline{c} \circ D_e \tag{1}$$

and, equivalently,

$$\bigwedge_{s\in S} d_s = case\{c.\overline{c} \circ D_e\}_{c:e\to s\in C} \tag{2}$$

(see section 10.3 and section 15.1).

For the definition of the type instance $D_e : e \to e[e_s/s \mid s \in S]$, see section 10.2.

If $e \in \mathcal{I}$, then $D_e = id_e$ and thus can be omitted in (1) and (2).

(1) is called an **inductive definition of D on $C\Sigma$**.

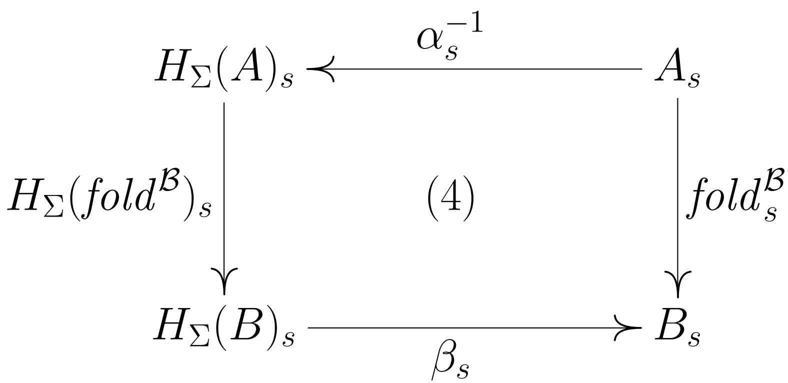
Proof. Let $\mathcal{B}$ be the $C\Sigma$-algebra that is defined as follows:

For all $s \in S$, $\mathcal{B}(s) = \mathcal{A}(e_s)$ and for all $c : e \to s$, $c^{\mathcal{B}} = \overline{c}^{\mathcal{A}}$.

Since $\mathcal{A}$ is initial in Alg_Σ, $fold^{\mathcal{B}} : \mathcal{A} \to \mathcal{B}$ is the unique Alg_{H_Σ}-morphism from $\alpha = ([c^{\mathcal{A}}]_{c:e\to s\in C})_{s\in S}$ to $\beta = ([c^{\mathcal{B}}]_{c:e\to s\in C})_{s\in S}$, i.e., $fold^{\mathcal{B}}$ is the unique S-sorted function such that the following diagram commutes for all $s \in S$:

$$\begin{array}{ccc}
H_\Sigma(A)_s & \xrightarrow{\ \alpha_s\ } & A_s \\
{\scriptstyle H_\Sigma(fold^{\mathcal{B}})_s}\downarrow & (3) & \downarrow{\scriptstyle fold^{\mathcal{B}}_s} \\
H_\Sigma(B)_s & \xrightarrow[\ \beta_s\ ]{} & B_s
\end{array}$$

By Lemma 4.2 (1), (3) commutes iff (4) commutes:



Define $g \in A^C$ by $g(d_s) = fold_s^{\mathcal{B}}$ for all $s \in S$.

(4) commutes iff g satisfies (2):

If (4) commutes, then for all $s \in S$,

$$g(d_s) = fold_s^{\mathcal{B}} \stackrel{(4)}{=} \beta_s \circ H_\Sigma(fold^{\mathcal{B}})_s \circ \alpha^{-1} = [c^{\mathcal{B}}]_{c:e\to s\in C} \circ (\textstyle\coprod_{c:e\to s\in C} fold_e^{\mathcal{B}}) \circ [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C}$$

$$\stackrel{(19)\ in\ chapter\ 2}{=} [c^{\mathcal{B}} \circ fold_e^{\mathcal{B}}]_{c:e\to s\in C} \circ [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C} = [\overline{c}^{\mathcal{A}} \circ fold_e^{\mathcal{B}}]_{c:e\to s\in C} \circ [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C}$$

$$= [\overline{c}^{\mathcal{A}} \circ D_e^{\mathcal{A}}]_{c:e\to s\in C} \circ [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C} = [(\overline{c} \circ D_e)^{\mathcal{A}}]_{c:e\to s\in C} \circ [c^{\mathcal{A}}]^{-1}_{c:e\to s\in C}$$

$$= (case\{c.\overline{c} \circ D_e\}_{c:e\to s\in C})^{\mathcal{A}},$$

i.e., g satisfies (2).

Conversely, if g satisfies (2), then for all $s \in S$,

$$fold_s^{\mathcal{B}} = g(d_s) \stackrel{(2)}{=} (case\{c.\overline{c} \circ D_e\}_{c:e \to s \in C})^{\mathcal{A}} = \dots \text{(see above)} \dots = \beta_s \circ H_\Sigma(fold^{\mathcal{B}})_s \circ \alpha^{-1},$$

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 14). In turn, paramorphisms are adjoint folds for an adjunction of the form $(\Delta^I : Set \to Set^I, \prod_{i \in I} : Set^I \to Set)$ (see chapter 25).

The proof of Theorem 16.1 also reveals that (1) is equivalent to the compatibility of d with the $C\Sigma$-homomorphism $fold^{\mathcal{B}} : \mathcal{A} \to \mathcal{B}$. Hence, basically, the unique solvability of (1) in $\mathcal{A}$ follows from the uniqueness of a $C\Sigma$-homomorphism from $\mathcal{A}$ to $\mathcal{B}$, which in turn follows from the initiality of $\mathcal{A}$ in $Alg_{C\Sigma}$.

Theorem 16.2

Let $D\Sigma = (S, D)$ be a destructive polynomial subsignature of Σ, $\mathcal{A}|_{D\Sigma}$ be final in $Alg_{D\Sigma}$ and $C = \{c_s : e_s \to s \mid s \in S\}$ be a set of polynomial constructors.

For all $d : s \to e \in D$, let $\overline{d} : e_s \to e[e_s/s \mid s \in S]$ be a closed $\lambda\Sigma$-term.

There is a unique solution $g \in A^C$ in $\mathcal{A}$ of the Σ-formulas

$$\bigwedge_{d:s\to e\in D} d \circ c_s = C_e \circ \overline{d} \tag{1}$$

and, equivalently,

$$\bigwedge_{s\in S} c_s = obj\{d.C_e \circ \overline{d}\}_{d:s\to e'\in D} \tag{2}$$

(see section 10.3 and section 15.2).

For the definition of the type instance $C_e : e[e_s/s \mid s \in S] \to e$, see section 10.2.

If $e \in \mathcal{I}$, then $C_e = id_e$ and thus can be omitted in (1) and (2).

(1) is called a **coinductive definition of C on $D\Sigma$**.

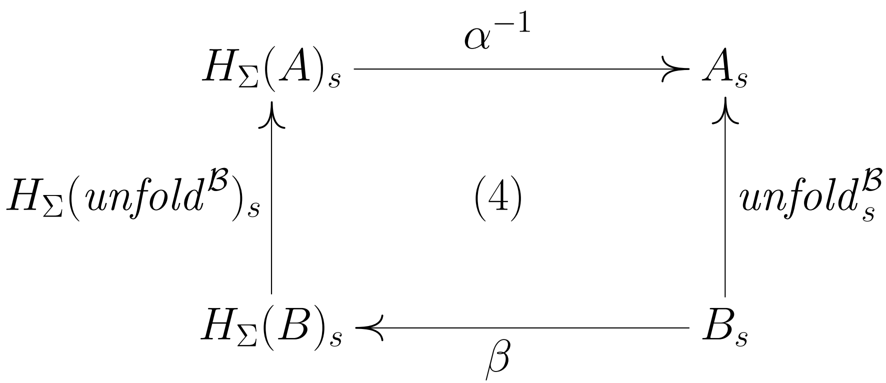
Proof. Let $\mathcal{B}$ be the $D\Sigma$-algebra that is defined as follows:

For all $s \in S$, $\mathcal{B}(s) = \mathcal{A}(e_s)$ and for all $d : s \to e$, $d^{\mathcal{B}} = \overline{d}^{\mathcal{A}}$.

Since $\mathcal{A}$ is final in Alg_{Σ}, $unfold^{\mathcal{B}} : \mathcal{B} \to \mathcal{A}$ is the unique Alg_{H_Σ}-morphism from $\beta = (\langle d^{\mathcal{B}} \rangle_{d:s \to e \in D})_{s \in S}$ to $\alpha = (\langle d^{\mathcal{A}} \rangle_{d:s \to e \in D})_{s \in S}$, i.e., $unfold^{\mathcal{B}}$ is the unique S-sorted function such that the following diagram commutes for all $s \in S$:

$$
\begin{array}{ccc}
H_\Sigma(A)_s & \xleftarrow{\quad \alpha \quad} & A_s \\
{\scriptstyle H_\Sigma(unfold^{\mathcal{B}})_s} \uparrow & (3) & \uparrow {\scriptstyle unfold^{\mathcal{B}}_s} \\
H_\Sigma(B)_s & \xleftarrow[\quad \beta \quad]{} & B_s
\end{array}
$$

By Lemma 4.2 (2), (3) commutes iff (4) commutes:



Define $g \in A^C$ by $g(c_s) = unfold^{\mathcal{B}}_s$ for all $s \in S$.

(4) commutes iff g satisfies (2):

If (4) commutes, then for all $s \in S$,

$$\begin{aligned}
&g(c_s) = unfold^{\mathcal{B}}_s \overset{(4)}{=} \alpha^{-1} \circ H_\Sigma(unfold^{\mathcal{B}})_s \circ \beta_s \\
&= \langle d^{\mathcal{A}} \rangle^{-1}_{d:s\to e\in D} \circ (\textstyle\prod_{d:s\to e\in D} unfold^{\mathcal{B}}_e) \circ \langle d^{\mathcal{B}} \rangle_{d:s\to e\in D} \\
&\overset{(8)\ in\ chapter\ 2}{=} \langle d^{\mathcal{A}} \rangle^{-1}_{d:s\to e\in D} \circ \langle unfold^{\mathcal{B}}_e \circ d^{\mathcal{B}} \rangle_{d:s\to e\in D} \\
&= \langle d^{\mathcal{A}} \rangle^{-1}_{d:s\to e\in D} \circ \langle unfold^{\mathcal{B}}_e \circ \overline{d}^{\mathcal{A}} \rangle_{d:s\to e\in D} = \langle d^{\mathcal{A}} \rangle^{-1}_{d:s\to e\in D} \circ \langle C^{\mathcal{A}}_e \circ \overline{d}^{\mathcal{A}} \rangle_{d:s\to e\in D} \\
&= \langle d^{\mathcal{A}} \rangle^{-1}_{d:s\to e\in D} \circ \langle (C_e \circ \overline{d})^{\mathcal{A}} \rangle_{d:s\to e\in D} = (obj\{d.C_e \circ \overline{d}\}_{d:s\to e\in D})^{\mathcal{A}},
\end{aligned}$$

i.e., g satisfies (2).

Conversely, if g satisfies (2), then for all $s \in S$,

$$unfold_s^{\mathcal{B}} = g(c_s) \stackrel{(2)}{=} (obj\{d.C_e \circ \overline{d}\}_{d:s\to e'\in D})^{\mathcal{A}} = \dots \text{(see above)} \dots$$
$$= \alpha^{-1} \circ H_\Sigma(unfold^{\mathcal{B}})_s \circ \beta,$$

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 14). In turn, apomorphisms are adjoint unfolds for an adjunction of the form $(\coprod_{i\in I} : Set^I \to Set, \Delta^I : Set \to Set^I)$ (see chapter 25).

The proof of Theorem 16.2 also reveals that (1) is equivalent to the compatibility of c with the $D\Sigma$-homomorphism $unfold^{\mathcal{B}}:\mathcal{B}\rightarrow\mathcal{A}$. Hence, basically, the unique solvability of (1) in $\mathcal{A}$ follows from the uniqueness of a $D\Sigma$-homomorphism from $\mathcal{B}$ to $\mathcal{A}$, which in turn follows from the finality of $\mathcal{A}$ in $Alg_{D\Sigma}$.

Theorem 16.2 may require that several functions are defined simultaneously (see sample coinductive definitions 16.4.8, 16.4.11, 16.4.12, 16.4.18, 16.4.23 and 16.4.29).

Coinductive definitions restrict the "recursive calls" of the functions to be defined to outermost term positions. The following theorem will provide a definitional schema that admits the proper embedding of recursive calls and does not require that several functions to be defined simultaneously are turned into their sum extension.

However, the new schema restricts the structure of terms on the right-hand sides of defining equations insofar as destructors may occur only at innermost (non-variable) term positions.

Theorem 16.3

Let $C\Sigma = (S', C')$ be a constructive polynomial signature of Σ, $D\Sigma = (S, D)$ be a destructive signature, $\Sigma = C\Sigma \cup D\Sigma$, $C = \{c \in C \mid trg(c) \in S\}$ and $\mathcal{A}|_{D\Sigma}$ be final in $Alg_{D\Sigma}$.

- For all $c : e_c \to s \in C$ and $d : s \to e \in D$, let $\prod_{i=1}^{n_c} s_{c,i} = e_c$, $t_{c,d} : e$ be a $\lambda\Sigma$-term over $V_c =_{def} \{x_{c,i} \mid 1 \leq i \leq n_c\} \subseteq V$, $x_c = (x_{c,1}, \dots, x_{c,n_c})$, $u \in T_{C\Sigma}(V)$ and σ be a $D\Sigma$-substitution such that $t_{c,d} = u\sigma$ and for all $x \in V_{u,\sigma} = var(u) \cap supp(\sigma)$, $\sigma(x)$ is flat.

There is a unique solution $g \in A^C$ of the Σ-formulas

$$\bigwedge_{c:e_c \to s \in C,\ d:s \to e \in D} \forall x_{c,1} \dots \forall x_{c,n_c}\ d(c(x_c)) = t_{c,d} : e \tag{1}$$

and, equivalently,

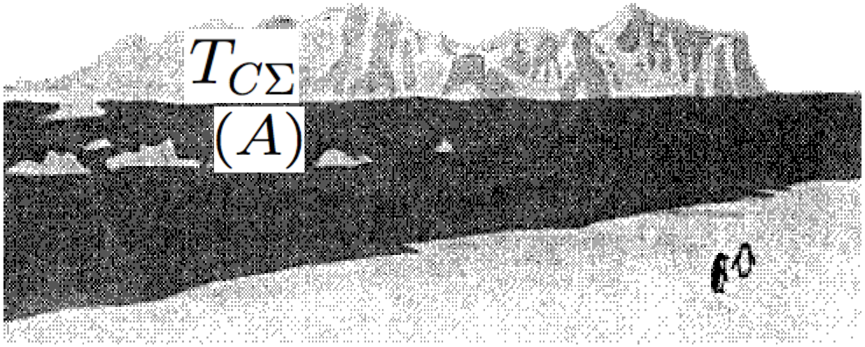
$$\bigwedge_{c:e_c \to s \in C,\ d:s \to e \in D} d \circ c = \lambda x_c.t_{c,d} : e_c \to e \tag{2}$$

and, equivalently,

$$\bigwedge_{c:e_c \to s \in C} c = obj\{d.\lambda x_c.t_{c,d} : e_c \to e\}_{d:s \to e \in D}$$

in $\mathcal{A}$ (see section 10.1 and section 15.2).

(1) is called a **biinductive definition of C on** $D\Sigma$.



Proof. Let $C\Sigma(A)$ be the term grounding of $C\Sigma$ on A (see section 9.12) and $\mathcal{B}$ be the initial $C\Sigma(A)$-algebra with carrier $T_{C\Sigma(A)}$ and the following interpretation of D:

For all $c : e_c \to s \in C$, $d : s \to e \in D$ and valuations g' of V_c in $T_{C\Sigma(A)}$ and $a \in A_s$,

$$d^{\mathcal{B}}(c(g'(x_{c,1}), \dots, g'(x_{c,n_c}))) =_{def} t^{\mathcal{B}}_{c,d}(g'), \quad (3)$$

$$d^{\mathcal{B}}(val_s(a)) =_{def} val^{\mathcal{B}}_e(d^{\mathcal{A}}(a)) \quad (4)$$

where $val^{\mathcal{B}}_e : A_e \to T_{C\Sigma(A),e}$ denotes the $\mathcal{T}_{fo}(S)$-extension of the S-sorted function

$$val^{\mathcal{B}} : A \to T_{C\Sigma(A)}$$

that maps $a \in A$ to the term $val_s(a)$.

By assumption, there are $u \in T_{C\Sigma}(V)$ and $D\Sigma$-substitution σ such that $t_{c,d} = u\sigma$ and for all $x \in V_{u,\sigma}$, $\sigma(x)$ is flat. Hence there are $1 \leq i_x \leq n_c$ and $d_x : s_{c,i_x} \to e \in Arr_\Sigma$ such that $\sigma(x) = d_x(x_{c,i_x})$ and $d_x \in D$ or $d_x = \pi_j \circ d$ for some $j \in \mathcal{I}$ and $d \in D$. Therefore,

$$t^{\mathcal{B}}_{c,d}(g') = (u\sigma)^{\mathcal{B}}(g') \stackrel{Prop.\ 10.1}{=} u^{\mathcal{B}}(\sigma^{\mathcal{B}}(g')) = u\{\sigma^{\mathcal{B}}(g')(x)/x \mid x \in V_{u,\sigma}\}$$
$$= u\{\sigma(x)^{\mathcal{B}}(g')/x \mid x \in V_{u,\sigma}\} = u\{d_x(g'(x_{c,i_x}))^{\mathcal{B}}/x \mid x \in V_{u,\sigma}\}$$
$$= u\{d^{\mathcal{B}}_x(g'(x_{c,i_x}))/x \mid x \in V_{u,\sigma}\}.$$

Since u consists of constructors and variables, we conclude that for all destructors or projections d and subterms $d(v)$ of $t_{c,d}$, $d(v)^{\mathcal{B}}_x(g') = d^{\mathcal{B}}(g'(x_{c,i_x}))$ for some $x \in V_{u,\sigma}$.

Therefore, (3) and (4) define $d^{\mathcal{B}}$ inductively on $T_{C\Sigma(A)}$.

Since $\mathcal{A}|_{D\Sigma}$ is final in $Alg_{D\Sigma}$, there is a unique $D\Sigma$-homomorphism $unfold^{\mathcal{B}} : \mathcal{B} \to \mathcal{A}$.

By (4), $val^{\mathcal{B}}$ and thus $unfold^{\mathcal{B}} \circ val^{\mathcal{B}} : A \to A$ are $D\Sigma$-homomorphic. Hence

$$unfold^{\mathcal{B}} \circ val^{\mathcal{B}} = id_A, \tag{5}$$

again because $\mathcal{A}|_{D\Sigma}$ is final in $Alg_{D\Sigma}$.

**** $\mathcal{A}$ is a $C\Sigma(A)$-algebra: For all $c : e_1 \times \cdots \times e_n \to s \in C$ and $s \in S$,

$$c^{\mathcal{A}} =_{def} unfold^{\mathcal{B}}_s \circ c^{\mathcal{B}} \circ val^{\mathcal{B}}_e, \tag{6}$$
$$val^{\mathcal{A}}_s =_{def} id_{A_s}. \tag{7}$$

(8) The greatest $D\Sigma$-congruence $\sim$ on $\mathcal{B}$ is a $C\Sigma$-congruence: By (3), $\mathcal{B}$ satisfies (2). Hence by Lemma 16.4, 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 13.3 (4), $ker(unfold^{\mathcal{B}})$ is the greatest $D\Sigma$-congruence on $\mathcal{B}$. Hence by (8), $ker(unfold^{\mathcal{B}})$ is a $C\Sigma$-congruence, i.e., for all $c : e \to s \in C$ and $t, t' \in T_{C\Sigma(A),e}$, $unfold^{\mathcal{B}}(t) = unfold^{\mathcal{B}}(t')$ implies $unfold^{\mathcal{B}}(c^{\mathcal{B}}(t)) = unfold^{\mathcal{B}}(c^{\mathcal{B}}(t'))$. Consequently and since by (5), $unfold^{\mathcal{B}} : T_{C\Sigma(A)} \to A$ is surjective, a $C\Sigma$-algebra $\mathcal{A}'$ with carrier A is defined as follows:

For all $c : e \to s \in C$ and $t, t' \in T_{C\Sigma(A),e}$,

$$c^{\mathcal{A}'}(unfold^{\mathcal{B}}(t)) =_{def} unfold^{\mathcal{B}}(c^{\mathcal{B}}(t)). \tag{9}$$

(10) $\mathcal{A}$ and $\mathcal{A}'$ interpret C in the same way: For all $c : e \to s \in C$ and $a \in A_e$,

$$c^{\mathcal{A}'}(a) \overset{(5)}{=} c^{\mathcal{A}'}(unfold^{\mathcal{B}}(val^{\mathcal{B}}(a))) \overset{(9)}{=} unfold^{\mathcal{B}}(c^{\mathcal{B}}(val^{\mathcal{B}}(a))) \overset{(6)}{=} c^{\mathcal{A}}(a).$$

(11) $unfold^{\mathcal{B}}$ is $C\Sigma(A)$-homomorphic: For all $c : e \to s \in C$ and $t \in T_{C\Sigma(A),e}$,

$$unfold^{\mathcal{B}}(c^{\mathcal{B}}(t)) \stackrel{(9)}{=} c^{\mathcal{A}'}(unfold^{\mathcal{B}}(t)) \stackrel{(10)}{=} c^{\mathcal{A}}(unfold^{\mathcal{B}}(t)).$$

For all $s \in S$ and $a \in A_s$,

$$unfold^{\mathcal{B}}(val_s^{\mathcal{B}}(a)) \stackrel{(5)}{=} a \stackrel{(7)}{=} val_s^{\mathcal{A}}(a) \stackrel{unfold^{\mathcal{B}}\ D\Sigma-hom.}{=} val_s^{\mathcal{A}}(unfold_{A_s}^{\mathcal{B}}(a)).$$

Define $g \in A^V$ by $g(c) = c^{\mathcal{A}}$ for all $c : e_c \to s \in C$.

$g \in A^V$ satisfies (1):

For all $d : s \to e \in D$, $c : \prod_{i=1}^{n_c} s_i \to s \in C$, $a = (a_1, \ldots, a_{n_c}) \in \prod_{i=1}^{n_c} A_{s_i}$ and the valuations $g', h' : V_c \to T_{C\Sigma(A)}$ with

$$g'(x_{c,i}) = val_{s_i}^{\mathcal{B}}(a_i) \quad \text{and} \quad h'(x_{c,i}) = a_i$$

for all $1 \leq i \leq n_c$,

$$d^{\mathcal{A}}(g(c)(a)) = d^{\mathcal{A}}(c^{\mathcal{A}}(a)) \stackrel{(6)}{=} d^{\mathcal{A}}(unfold_s^{\mathcal{B}}(c^{\mathcal{B}}(val^{\mathcal{B}}(a)))) = d^{\mathcal{A}}(unfold_s^{\mathcal{B}}(c(val^{\mathcal{B}}(a))))$$

$$\stackrel{unfold^{\mathcal{B}}\ D\Sigma-hom.}{=} unfold_e^{\mathcal{B}}(d^{\mathcal{B}}(c(val^{\mathcal{B}}(a)))) \stackrel{(3)}{=} unfold_e^{\mathcal{B}}(t_{c,d}^{\mathcal{B}}(g'))$$

$$\stackrel{(11),\ unfold^{\mathcal{B}}\ D\Sigma-hom.}{=} t_{c,d}^{\mathcal{A}}(unfold^{\mathcal{B}} \circ g') = t_{c,d}^{\mathcal{A}}(unfold^{\mathcal{B}} \circ val^{\mathcal{B}} \circ h') \stackrel{(5)}{=} t_{c,d}^{\mathcal{A}}(h').$$

(12) *unfold*$^{\mathcal{B}}$ agrees with the unique $C\Sigma(A)$-homomorphism $fold'^{\mathcal{A}} : \mathcal{B} \to \mathcal{A}$: Since $\mathcal{A}$ is $C\Sigma(A)$-algebra and $\mathcal{B}$ is initial in $Alg_{C\Sigma(A)}$, (12) follows from (11).

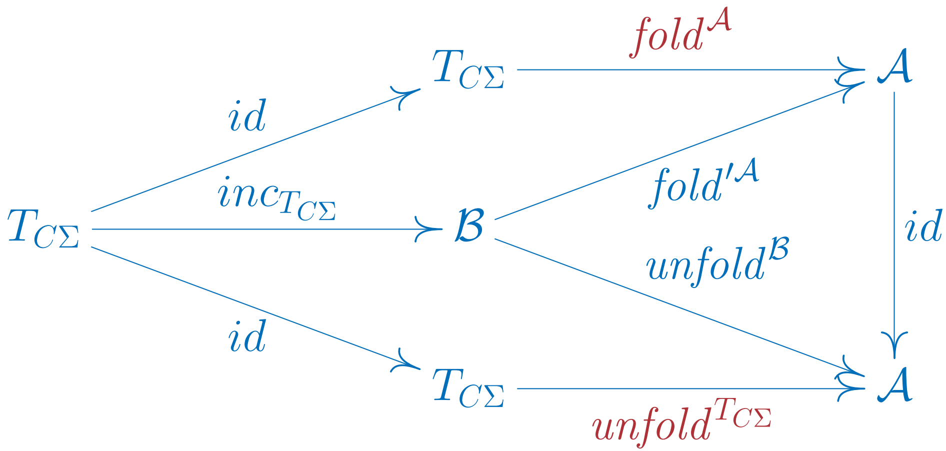
(13) *unfold*$^{T_{C\Sigma}}$ agrees with the unique $C\Sigma$-homomorphism $fold^{\mathcal{A}} : T_{C\Sigma} \to \mathcal{A}$: Since $inc_{T_{C\Sigma}}$ and $fold'^{\mathcal{A}}$ are $C\Sigma$-homomorphisms and $T_{C\Sigma}$ is initial in $Alg_{C\Sigma}$,

$$fold^{\mathcal{A}} = fold'^{\mathcal{A}} \circ inc_{T_{C\Sigma}}. \tag{14}$$

By (3) and the condition on $t_{c,d}$, for all $c : e_c \to s \in C$, $d : s \to e \in D$, $t \in T_{C\Sigma,e_c}$ and $a \in A_s$, $d^{\mathcal{B}}(c(t)) = (\lambda x_c.t_{c,d})^{\mathcal{B}}(t) \in T_{C\Sigma,e}$. Hence $T_{C\Sigma}$ is a $D\Sigma$-invariant of $\mathcal{B}$ and thus $inc_{T_{C\Sigma}} : T_{C\Sigma} \to \mathcal{B}$ is $D\Sigma$-homomorphic. Therefore,

$$unfold^{T_{C\Sigma}} = unfold^{\mathcal{B}} \circ inc_{T_{C\Sigma}}. \tag{15}$$

(12), (14) and (15) imply (13).



It remains to show that every solution h of (1) in $\mathcal{A}$ agrees with g.

Let $\sim$ be the least S-sorted equivalence relation on A such that $\sim$ contains Δ_A and for all $c : e \to s \in C$ 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 on $\mathcal{A}$.

For all $c : e \to s \in C$ 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 13.3 (3), Δ_A is the only $D\Sigma$-congruence on $\mathcal{A}$. Hence $g = h$.

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 $d : s \to e \in D$.

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 $c : \prod_{i=1}^n s_{c,i} \to s \in C$, $a' = (a_1, \dots, a_n), b' = (b_1, \dots, b_n) \in \prod_{i=1}^n A_{s_i}$ such that $a = c^{\mathcal{A}}(a')$, $b = c^{\mathcal{A}}(b')$ and $a' \sim b'$. Then for all $1 \leq i \leq n$, $a_i \sim b_i$.

Let $g', h' : \{x_1, \ldots, x_n\} \to A$ be the valuations such that for all $1 \leq i \leq n$, $g'(x_i) = a_i$ and $h'(x_i) = b_i$. By assumption, there are $u \in T_{C\Sigma}(V)$ and $D\Sigma$-substitution σ such that $t_{c,d} = u\sigma$ and for all $x \in V_{u,\sigma}$, $\sigma(x)$ is flat. Hence there are $1 \leq i_x \leq n$ and $d_x : s_{i_x} \to e \in Arr_\Sigma$ such that $\sigma(x) = d_x(x_{i_x})$ and $d_x \in D$ or $d_x = \pi_j \circ d$ for some $j \in \mathcal{I}$ and $d \in D$. Therefore, $\sigma^{\mathcal{A}}(g')(x) = \sigma(x)^{\mathcal{A}}(g') = d_x^{\mathcal{A}}(g'(x_{i_x})) = d_x^{\mathcal{A}}(a_{i_x})$ and analogously, $\sigma^{\mathcal{A}}(h')(x) = d_x^{\mathcal{A}}(b_{i_x})$. Hence by induction hypothesis, for all $x \in V_{u,\sigma}$,

$$\sigma^{\mathcal{A}}(g')(x) = d_x^{\mathcal{A}}(a_i) \sim d_x^{\mathcal{A}}(b_i) = \sigma^{\mathcal{A}}(h')(x),$$

and thus by the definition of $\sim$,

$$u^{\mathcal{A}}(\sigma^{\mathcal{A}}(g')) \sim u^{\mathcal{A}'}(\sigma^{\mathcal{A}}(h')) \tag{16}$$

where $\mathcal{A}'$ denotes the Σ-algebra such that $\mathcal{A}'|_{D\Sigma} = \mathcal{A}|_{D\Sigma}$ and for all $c \in C$, $c^{\mathcal{A}'} = h(c)$.

Since g and h satisfy (1) and for all $c \in C$, $c^{\mathcal{A}} = g(c)$, we finally obtain

$$d^{\mathcal{A}}(a) = t_{c,d}^{\mathcal{A}}(g') = (u\sigma)^{\mathcal{A}}(g') \stackrel{Prop.\ 10.1}{=} u^{\mathcal{A}}(\sigma^{\mathcal{A}}(g')) \stackrel{(16)}{\sim} u^{\mathcal{A}'}(\sigma^{\mathcal{A}}(h')) = u^{\mathcal{A}'}(\sigma^{\mathcal{A}'}(h'))$$
$$\stackrel{Prop.\ 10.1}{=} (u\sigma)^{\mathcal{A}'}(h') = t_{c,d}^{\mathcal{A}'}(h') = d^{\mathcal{A}}(b).$$

❑

Crucial arguments in the proof of Theorem 16.3 originate from the proof of [157], Theorem 3.1, and [167], Theorem 2, which show that certain stream (resp. infinite-bintree) equations have unique solutions in final stream (resp. infinite-bintree) algebras and thus justifies their designation as *definitions* of stream constructors (see also [74], Appendices A.5 and A.6; [93], section 4; [66], section 8.2).

In a rather informal way, [159], 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 [167], section 3, [34], [67], [78, 153, 81] and [157], section 9; respectively.

Theorem 16.3 provides the coincidence of a *fold* and an *unfold* (see (12) in the proof), which is often regarded as a correspondence between denotational semantics and operational semantics (e.g., in [78]).

Consequently, biinductive definitions are closely related to structural-operational (SOS) rules, which, e.g., determine the syntax (constructors) and operational semantics of programming languages and 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 (in applicative form; see sample algebra 9.6.23 and sample biinductive definition 16.5.5): Let $B \in \mathcal{P}_+(X)$, $x \in X$ and $c \in 2$.

$$\frac{t \xrightarrow{\delta,x} t',\ \ u \xrightarrow{\delta,x} u'}{t + u \xrightarrow{\delta,x} t' + u'} \qquad \frac{t \xrightarrow{\delta,x} t',\ \ u \xrightarrow{\delta,x} u'}{t * u \xrightarrow{\delta,x} t' * u + \widehat{\beta(t)} * u'}$$

$$\frac{t \xrightarrow{\delta,x} t'}{star(t) \xrightarrow{\delta,x} t' * star(t)} \qquad \frac{}{\widehat{c} \xrightarrow{\delta,x} \widehat{0}} \qquad \frac{}{B \xrightarrow{\delta,x} \widehat{x \in B}}$$

$$\frac{t \xrightarrow{\beta} m,\ \ u \xrightarrow{\beta} n}{t + u \xrightarrow{\beta} max(m,n)} \qquad \frac{t \xrightarrow{\beta} m,\ \ u \xrightarrow{\beta} n}{t * u \xrightarrow{\beta} m * n}$$

$$\frac{}{star(t) \xrightarrow{\beta} 0} \qquad \frac{}{\overline{B} \xrightarrow{\beta} 0} \qquad \frac{}{\widehat{c} \xrightarrow{\beta} c}$$

[26, 37, 81, 91, 177] present biinductive definitions in a more category-theoretical form, which, instead of employing the term algebra $\mathcal{B}$ of Theorem 16.3, involves distributive laws (see chapter 23). Unfortunately, adapting non-trivial sets of defining equations to this framework may be a quite difficult task—as the simple example given in chapter 24 suggests. In this context, some notions differ from ours. For instance, [26] would call coinductive definitions *coiterative* and biinductive ones λ-*coiterative*.

16.2 Bisimulation modulo constructors

Let $D\Sigma = (S, D)$ be a destructive signature, $C\Sigma = (S', C')$ be a constructive signature, $C = \{c \in C' \mid trg(c) \in S\}$, $\Sigma = C\Sigma \cup D\Sigma$ and $\mathcal{A}$ be a Σ-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 **$D\Sigma$-bisimulation modulo C on $\mathcal{A}$** if for all $d : s \to e \in D$ and $a, b \in A_s$,

$$a \sim_s b \quad \Rightarrow \quad d^{\mathcal{A}}(a) \sim_{C,e} d^{\mathcal{A}}(b).$$

Lemma 16.4

Suppose that the assumptions of Theorem 16.3 hold true and $\sim$ is 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 $d : s \to e \in D$.

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 $c:\prod_{i=1}^n s_{c,i}\to s\in C$, $a'=(a_1,\dots,a_n), b'=(b_1,\dots,b_n)\in\prod_{i=1}^n A_{s_i}$ such that $a=c^{\mathcal{A}}(a')$, $b=c^{\mathcal{A}}(b')$ and $a'\sim_C b'$. Then for all $1\le i\le n$, $a_i\sim_C b_i$.

Let $g',h':\{x_1,\dots,x_n\}\to A$ be the valuations such that for all $1\le i\le n$, $g'(x_i)=a_i$ and $h'(x_i)=b_i$. By assumption, $t_{c,d}=u\sigma$ for some $u\in T_{C\Sigma}(V)$ and a $D\Sigma$-substitution σ and for all $x\in var(u)\cap supp(\sigma)$, $\sigma(x)$ is flat, there are $d_x\in D$ and $1\le i_x\le n$ such that $\sigma(x)=d_x(x_{c,i_x})$ and thus $\sigma^{\mathcal{A}}(g')(x)=\sigma(x)^{\mathcal{A}}(g')=d_x^{\mathcal{A}}(g'(x_{i_x}))=d_x^{\mathcal{A}}(a_{i_x})$ and analogously, $\sigma^{\mathcal{A}}(h')(x)=d_x^{\mathcal{A}}(b_{i_x})$.

Hence by induction hypothesis, for all $x\in var(u)\cap supp(\sigma)$,

$$\sigma^{\mathcal{A}}(g')(x)=d_{i_x}^{\mathcal{A}}(a_i)\sim_C d_{i_x}^{\mathcal{A}}(b_i)=\sigma^{\mathcal{A}}(h')(x),$$

and thus by the definition of $\sim_C$,

$$u^{\mathcal{A}}(\sigma^{\mathcal{A}}(g'))\sim_C u^{\mathcal{A}}(\sigma^{\mathcal{A}}(h')). \tag{1}$$

Since the valuation $\lambda c.c^{\mathcal{A}}:C\to A$ satisfies Theorem 16.3(1),

$$d^{\mathcal{A}}(a)=t_{c,d}^{\mathcal{A}}(g')=(u\sigma)^{\mathcal{A}}(g')\overset{Prop.\ 10.1}{=}u^{\mathcal{A}}(\sigma^{\mathcal{A}}(g'))\overset{(1)}{\sim_C}u^{\mathcal{A}}(\sigma^{\mathcal{A}}(h'))\overset{Prop.\ 10.1}{=}(u\sigma)^{\mathcal{A}}(h')$$
$$=t_{c,d}^{\mathcal{A}}(h')=d^{\mathcal{A}}(b).$$ ❏

16.3 Sample inductive definitions

Let the assumption of Theorem 16.1 hold true. It tells us that an inductive definition

$$\bigwedge_{c:e\to s\in C} d_s \circ c = \overline{c} \circ D_e \quad (1)$$

of $D = \{d_s : s \to e_s \mid s \in S\}$ of polynomial destructors has a unique solution g in $\mathcal{A}$.

Often the functions that satisfy (1) are product extensions of the functions of E and a further function f. If f is an identity, then E specifies a primitive recursive function or paramorphism (see chapter 14).

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 $D_e : e \to e[e_s/s \mid s \in S]$. In turn, this requires a general definition of $\overline{c}$, not only for the D-images where $\overline{c}$ is applied to in the equations.

In [79], the formation of a definition of $\overline{c}$ 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 $\overline{c}$ is presented as a $\lambda\Sigma$-term.

Inductively defined functions on natural numbers

Let $C\Sigma = Nat$ (see section 8.2). For $D = \{d : nat \to e\}$, (1) reads as follows:

$$d \circ zero = \overline{zero} : 1 \to e, \tag{2}$$

$$d \circ succ = \overline{succ} \circ d : nat \to e. \tag{3}$$

Let $z : 1$, $b : 2$, $k, m, n : nat$, $f : nat \to nat \in V$, $one = succ \circ zero$ and $(\leq) : nat \times nat \to 2$ be interpreted in $\mathcal{A}|_{Nat}$ as usually.

1. Let $d : nat \to 2$ denote the test on zero with the equations

$$\begin{aligned} d(zero(z)) &= 1, \\ d(succ(n)) &= 0. \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ zero &= \overline{1}, \\ d \circ succ &= (\lambda b.0) \circ d. \end{aligned}$$

2. Let $pred : nat \to nat$ denote the predecessor function with the equations

$$\begin{aligned} pred(zero(z)) &= zero(z), \\ pred(succ(n)) &= n. \end{aligned}$$

Let $d : nat \to nat \times nat$ denote the product extension $\langle pred, id\rangle$ with the equations

$$\begin{aligned} d(zero(z)) &= (zero(z), zero(z)), \\ d(succ(k)) &= (\lambda(m,n).(n, succ(n)))(d(k)) \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ zero &= \langle zero(z), zero(z)\rangle, \\ d \circ succ &= (\lambda(m,n).(n, succ(n))) \circ d. \end{aligned}$$

Hence

$$g[(\pi_1 \circ g(d))/pred] \tag{4}$$

uniquely solves the above *pred*-equations and $\pi_1 \circ g(d)$ is a paramorphism (see chapter 14).

3. Let $add : nat \times nat \to nat$ denote the addition of natural numbers with the equations

$$\begin{aligned} add(zero(z), n) &= n, \\ add(succ(m), n) &= succ(add(m,n)). \end{aligned}$$

Let $d : nat \to (nat \to nat) \in V$ denote $curry(add)$ with the equations

$$\begin{aligned} d(zero(z)) &= id_{nat}, \\ d(succ(m)) &= \lambda n.succ(d(m)(n)) = (\lambda f.\lambda n.succ(f(n)))(d(m)). \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ zero &= \lambda z.id_{nat}, \\ d \circ succ &= (\lambda m.\lambda f.\lambda n.succ(f(n))) \circ d. \end{aligned}$$

Hence $g[uncurry(g(d))/add]$ uniquely solves the above add-equations.

4. ([180], Example 12 - a *histomorphism*; [72], Example 21) Let $fib : nat \to nat$ denote the Fibonacci function with the equations

$$\begin{aligned} fib(zero(z)) &= zero(z), \\ fib(succ(zero(z))) &= one(z), \\ fib(succ(succ(n))) &= add(fib(n), fib(succ(n))). \end{aligned}$$

Let $d : nat \to nat \times nat$ denote the product extension $\langle fib, fib \circ succ \rangle$ with the equations

$$\begin{aligned} d(zero(z)) &= (zero(z), one(z)), \\ d(succ(k)) &= (\lambda(m,n).(n, add(m,n)))(d(k)). \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ zero &= \langle zero, one \rangle, \\ d \circ succ &= (\lambda(m,n).(n, add(m,n))) \circ d. \end{aligned}$$

Hence $g[(\pi_1 \circ g(d))/fib]$ uniquely solves the above fib-equations.

5. Let $mul : nat \times nat \to nat$ denote the multiplication of natural numbers with the equations

$$\begin{aligned} mul(zero(z), n) &= zero(z), \\ mul(succ(m), n) &= add(mul(m, n), n). \end{aligned}$$

Let $d : nat \to (nat \to nat) \in V$ denote $curry(mul)$ with the equations

$$\begin{aligned} d(zero(z)) &= \lambda n.zero(z), \\ d(succ(m)) &= \lambda n.add(d(m)(n), n) \;=\; (\lambda f.\lambda n.add(f(n), n))(d(m)). \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ zero &= \lambda z.\lambda n.zero(z), \\ d \circ succ &= (\lambda f.\lambda n.add(f(n), n)) \circ d. \end{aligned}$$

Hence $g[uncurry(g(d))/mul]$ uniquely solves the above mul-equations.

6. ([72], Example 20) Let $fact : nat \to nat$ denote the factorial function with the equations

$$\begin{aligned} fact(zero(z)) &= one(z), \\ fact(succ(n)) &= mul(fact(n), succ(n)). \end{aligned}$$

Let $d : nat \to nat \times nat$ denote the product extension $\langle fact, id\rangle$ with the equations

$$\begin{aligned} d(zero(z)) &= (one(z), zero(z)), \\ d(succ(k)) &= (\lambda(m,n).(mul(m, succ(n)), succ(n)))(d(k)). \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ zero &= \langle one, zero\rangle, \\ d \circ succ &= (\lambda(m,n).(mul(m, succ(n)), succ(n))) \circ d. \end{aligned}$$

Hence $g[(\pi_1 \circ g(d))/fact]$ uniquely solves the above $fact$-equations and $\pi_1 \circ g(d)$ is a paramorphism (see chapter 14).

7. Let $sub : nat \times nat \to nat$ denote the subtraction on natural numbers with the equations

$$\begin{aligned} sub(zero(z), n) &= zero(z), \\ sub(succ(m), zero(z)) &= succ(m), \\ sub(succ(m), succ(n)) &= sub(m, n). \end{aligned}$$

Let $d : nat \to (nat \to nat) \times nat$ denote the product extension $\langle curry(sub), id_{nat} \rangle$ with the equations

$$\begin{array}{rcl} d(zero(z)) & = & (\lambda n.zero(z), zero(z)), \\ d(succ(k)) & = & (\lambda(f,m).(case\{zero.\lambda z.succ(m), succ.f\}, succ(m))\})(d(k)). \end{array}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{array}{rcl} d \circ zero & = & \langle \lambda z.\lambda n.zero(z), zero(z) \rangle, \\ d \circ succ & = & (\lambda(f,m).(case\{zero.\lambda z.succ(m), succ.f\}, succ(m))) \circ d. \end{array}$$

Hence $g[uncurry(\pi_1 \circ g(d))/sub]$ uniquely solves the above sub-equations.

8. Let $ack : nat \times nat \to nat$ denote the Ackermann function with the equations

$$\begin{array}{rcl} ack(zero(z), n) & = & succ(n) \\ ack(succ(m), zero(z)) & = & ack(m, one(z)) \\ ack(succ(m), succ(n)) & = & ack(m, ack(succ(m), n)). \end{array}$$

Let $d : nat \to (nat \to nat) \in V$ denote $curry(ack)$ with the equations

$$d(zero(z)) = succ \tag{5}$$

$$d(succ(m))(zero(z)) = d(m)(one(z)) \tag{6}$$

$$d(succ(m))(succ(n)) = d(m)(d(succ(m))(n)). \tag{7}$$

Similarly to [79], section 5.2, (6) and (7) can be generalized as follows: For all $f : \mathbb{N} \to \mathbb{N}$,

$$\begin{aligned} d_f \circ zero &= f \circ one, & (8) \\ d_f \circ succ &= f \circ d_f & (9) \end{aligned}$$

is an inductive definition of $d_f : nat \to nat \in V$ and thus has a unique solution $g_f : \mathbb{N} \to \mathbb{N}$. Let

$$\begin{aligned} \overline{succ} : (\mathbb{N} \to \mathbb{N}) &\to (\mathbb{N} \to \mathbb{N}) \\ f &\mapsto g_f. \end{aligned}$$

Hence the equations

$$\begin{aligned} d \circ zero &= \lambda z.succ & (10) \\ d \circ succ &= \overline{succ} \circ d & (11) \end{aligned}$$

form an inductive definition of d. Its unique solution $h : \mathbb{N} \to (\mathbb{N} \to \mathbb{N})$ solves (5)-(7):

$$h(zero(z)) \overset{(10)}{=} succ,$$

$$h(succ(m))(zero(z)) \overset{(11)}{=} \overline{succ}(h(m))(zero(z)) = g_{h(m)}(zero(z)) \overset{(8)}{=} h(m)(one(z)),$$

$$h(succ(m))(succ(n)) \overset{(11)}{=} \overline{succ}(h(m))(succ(n)) = g_{h(m)}(succ(n)) \overset{(9)}{=} h(m)(g_{h(m)}(n))$$

$$= h(m)(\overline{succ}(h(m))(n)) \overset{(11)}{=} h(m)(h(succ(m))(n)).$$

Therefore, $g[uncurry(g(d))/ack]$ uniquely solves the above ack-equations.

9. Let $x : X,\ f : X \to X^*,\ g : X^* \to X^*,\ h : X^* \to X^* \in V$ and

$$\begin{aligned} &replicate : nat \to (X \to X^*),\\ &take : nat \to (X^* \to 1 + X^*),\\ &takeStream : nat \to (X^{\mathbb{N}} \to X^*) \end{aligned}$$

be variables denoting the synonymous Haskell functions.

Inductive definitions of these functions read as follows:

$$\begin{aligned} replicate \circ zero &= \lambda z.\lambda x.\epsilon,\\ replicate \circ succ &= (\lambda f.\lambda x.(x \cdot f(x))) \circ replicate,\\ take \circ zero &= \lambda z.\lambda x.\iota_2(\epsilon),\\ take \circ succ &= (\lambda g.\lambda s.(case\{\alpha.\iota_1, cons.\lambda(x, s').(x \cdot g(s'))\})^{X^*}) \circ take,\\ takeStream \circ zero &= \lambda z.\lambda x.\epsilon,\\ takeStream \circ succ &= (\lambda h.\lambda s.(head^{InfSeq}(s) \cdot h(tail^{InfSeq}(s)))) \circ takeStream \end{aligned}$$

(see sample algebra 9.6.3 and 9.6.5).

The interpretation of the case distinction in X^* is well-defined because X^* is initial in $Alg_{List(X)}$ (see above).

Inductively defined functions on lists

Let X, Y, Z be sets and $C\Sigma = List(X)$. Suppose that Σ includes

$$\begin{aligned} List'(Y) &= List(Y)[state'/state, \alpha'/\alpha, cons'/cons], \\ List''(Z) &= List(Z)[state''/state, \alpha''/\alpha, cons''/cons] \end{aligned}$$

(see section 8.2) and $\mathcal{A}|_{List'(Y)}$ and $\mathcal{A}|_{List''(Z)}$ are initial in $Alg_{List'(Y)}$ and $Alg_{List''(Z)}$, respectively

For $D = \{d : state \to e\}$, (1) reads as follows:

$$d \circ \alpha = \overline{\alpha} : 1 \to e, \tag{12}$$

$$d \circ cons = \overline{cons} \circ (id_X \times d) : X \times state \to e. \tag{13}$$

Let $z : 1,\ b : 2,\ p : 2^{\mathbb{N}},\ x : X,\ y : Y,\ m, n : \mathbb{N},\ s, s' : state \in V$ and $(\leq) : \mathbb{N}^2 \to 2$ be defined as usually.

10. Let $d = length : state \to \mathbb{N}$ denote the synonymous Haskell function on A_{state} with the equations

$$\begin{aligned} d(\alpha(z)) &= zero(z), \\ d(cons(x, s)) &= succ(d(s)). \end{aligned}$$

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 $d = sum : state \to \mathbb{N}$ denote the synonymous Haskell function on A_{state} with the equations

$$
\begin{aligned}
d(\alpha(z)) &= zero(z), \\
d(cons(n, s)) &= add(n, d(s)).
\end{aligned}
$$

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 $conc : state \times state \to state$ denote the concatenation of lists with the equations

$$
\begin{aligned}
conc(\alpha(z), s) &= s \\
conc(cons(x, s), s') &= cons(x, conc(s, s')).
\end{aligned}
$$

Let $d : state \to (state \to state) \in V$ denote $curry(cons)$ with the equations

$$
\begin{aligned}
d(\alpha(z))(s) &= s, \\
d(cons(x, s))(s') &= cons(x, d(s)(s')).
\end{aligned}
$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ \alpha &= \lambda z.id_{state}, \\ d \circ cons &= (\lambda(x,f).\lambda s'.cons(x,f(s'))) \circ (id_X \times d) \end{aligned}$$

where $f : state \to state \in V$. Hence $g[uncurry(g(d))/conc]$ uniquely solves the above $conc$-equations.

13. Let $s' : state'$, $f : Y^X$, $g : Y^X$, $h : Y^X \to state'$, $h' : Z^{X \times Y} \to (state' \to state'') \in V$ and

$$\begin{aligned} &map : Y^X \to (state \to state'), \\ &zipWith : Z^{X \times Y} \to (state \to (state' \to state'')) \end{aligned}$$

be variables for the synonymous Haskell functions with the equations

$$\begin{aligned} map(f)(\alpha(z)) &= \alpha', \\ map(f)(cons(x,s)) &= cons'(f(x), map(f)(s)), \\ zipWith(g)(\alpha(z))(s') &= \alpha'', \\ zipWith(g)(s)(\alpha'(z)) &= \alpha'', \\ zipWith(g)(cons(x,s))(cons(y,s')) &= cons''(g(x,y), zipWith(g)(s)(s')). \end{aligned}$$

Let

$$\begin{aligned} &d : state \to (Y^X \to state'), \\ &d' : state \to (Z^{X \times Y} \to (state' \to state'')) \end{aligned}$$

be variables denoting *flip*(*map*) and *flip*(*zipWith*), respectively, with the equations

$$\begin{array}{rcl} d(\alpha(z))(f) &=& \alpha', \\ d(cons(x,s))(f) &=& cons'(f(x), d(s)(f)), \\ d'(\alpha(z))(g)(s') &=& \alpha'', \\ d'(cons(x,s))(g)(\alpha'(z)) &=& \alpha'', \\ d'(cons(x,s))(g)(cons'(y,s')) &=& cons''(g(x,y), d'(s)(g)(s')). \end{array}$$

$\mathcal{A}$-equivalent inductive definitions for d and d' read as follows:

$$\begin{array}{rcl} d \circ \alpha &=& \lambda z.\lambda f.\alpha', \\ d \circ cons &=& (\lambda(x,h).\lambda f.cons'(f(x), h(f))) \circ (id_X \times d), \\ d' \circ \alpha &=& \lambda z.\lambda g.\lambda s'.\alpha'', \\ d' \circ cons &=& (\lambda(x,h').\lambda g.case\{\alpha'.\alpha'', cons'.\lambda(y.s').cons''(g(x,y), h'(g)(s'))\}) \\ && \circ\ (id_X \times d'). \end{array}$$

Hence $g[flip(g(d))/map][flip(g(d'))/zipWith]$ uniquely solves the above equations for *map* and *zipWith*.

14. Let $p : 2^X \in V$ and $filter : 2^X \to (state \to state)$ denote the synonymous Haskell function with the equations

$$\begin{aligned} filter(p)(\alpha(z)) &= \alpha \\ filter(p)(cons(x,s)) &= ite(p(b), cons(x, filter(p)(s)), filter(p)(s)). \end{aligned}$$

Let $d : state \to (2^X \to state) \in V$ denote $flip(filter)$ with the equations

$$\begin{aligned} d(\alpha(z))(p) &= \alpha, \\ d(cons(x,s))(p) &= ite(p(x), cons(x, d(s)(p)), d(s)(p)). \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ \alpha &= \lambda z.\lambda p.\alpha, \\ d \circ cons &= (\lambda(x,f).\lambda p.ite(p(x), cons(x, f(p)), f(p))) \circ (id_X \times d) \end{aligned}$$

where $f : 2^X \to state \in V$.

Hence $g[flip(g(d))/filter]$ uniquely solves the above $filter$-equations.

15. Let $f : Y^{Y\times X} \in V$ and $foldl : Y^{Y\times X} \times Y \to (state \to Y)$ be a variable denoting the synonymous Haskell function with the equations

$$\begin{aligned} foldl(f,y)(\alpha(z)) &= y \\ foldl(f,y)(cons(x,s)) &= foldl(f,f(y,x))(s). \end{aligned}$$

Let $d : state \to (Y^{Y\times X} \times Y \to Y) \in V$ denote $flip(foldl)$ with the equations

$$\begin{aligned} d(\alpha(z))(f,y) &= y \\ d(cons(x,s))(f,y) &= d(s)(f,f(y,x)). \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ \alpha &= \lambda z.\pi_2, \\ d \circ cons &= (\lambda(x,g).\lambda(f,y).g(f,f(y,x))) \circ (id_X \times d) \end{aligned}$$

where $g : Y^{Y\times X} \times Y \to Y \in V$.

Hence $g[flip(g(d))/foldl]$ uniquely solves the above $foldl$-equations.

16. Let $f : Y^{X\times Y} \in V$ and $foldr : Y^{X\times Y} \times Y \to (state \to Y)$ be a variable denoting the synonymous Haskell function with the equations

$$\begin{aligned} foldr(f,y)(\alpha(z)) &= y \\ foldr(f,y)(cons(x,s)) &= f(x, foldr(f,y)(s)). \end{aligned}$$

Let $d : state \to (Y^{X\times Y} \times Y \to Y) \in V$ denote $flip(foldr)$ with the equations

$$\begin{aligned} d(\alpha(z))(f,y) &= y \\ d(cons(x,s))(f,y) &= f(x, d(s)(f,y)). \end{aligned}$$

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 $g : Y^{X\times Y} \times Y \to Y \in V$.

Hence $g[flip(g(d))/foldr]$ uniquely solves the above $foldr$-equations.

By the way, for all $List(X)$-algebras $\mathcal{B}$,

$$foldr^{\mathcal{A}}(cons^{\mathcal{B}}, \alpha^{\mathcal{B}}) = fold^{\mathcal{B}}$$

(see section 9.11).

17. Let $X = \mathbb{N}$ and $sorted : state \to 2$ denote the test on the sortedness of a list of natural numbers with the equations

$$\begin{aligned} sorted(\alpha(z)) &= 1 \\ sorted(cons(m, \alpha(z))) &= 1 \\ sorted(cons(m, cons(n, s))) &= m \leq n \wedge sorted(cons(n, s)). \end{aligned}$$

Let $d : state \to 2^{\mathbb{N}} \in V$ denote $\lambda s.\lambda n.sorted(cons(n, s))$ with the equations

$$\begin{aligned} d(\alpha(z)) &= \lambda n.1 \\ d(cons(m, s)) &= (\lambda p.\lambda n.m \leq n \wedge p(n))(d(s)). \end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ \alpha &= \lambda z.\lambda n.1, \\ d \circ cons &= (\lambda(m, p).\lambda n.m \leq n \wedge p(n)) \circ (id_{\mathbb{N}} \times d). \end{aligned}$$

Let $d' : state \to (2 \times state)$ denote the product extension $\langle sorted, id \rangle$ with the equations

$$\begin{aligned} d' \circ \alpha &= \langle \overline{1}, \alpha \rangle \\ d' \circ cons &= (\lambda(m, (b, s)).(d(s)(m), cons(m, s))) \circ (id_{\mathbb{N}} \times d'). \end{aligned}$$

Hence, $g[(\pi_1 \circ g(d'))/sorted]$ uniquely solves the above $sorted$-equations and $\pi_1 \circ g(d')$ is a paramorphism (see chapter 14).

18. Suppose that Σ includes $Bintree(L)$ and $\mathcal{A}|_{Bintree(L)}$ is initial in $Alg_{Bintree(L)}$ and thus isomorphic to $FBin(L)$ (see sample algebra 9.6.10).

Let $d = subtree : state \to (btree \to btree)$ denote the function that maps a node $s \in 2^*$ of a finite binary tree t with node labels from L to the subtree of t with root s.

Let $b : 2,\ lab : L,\ t, u, u' : btree,\ f : btree \to btree \in V$. d is specified by the equations

$$\begin{aligned} d(\alpha(z))(t) &= t, & (1)\\ d(cons(b,s))(bjoin(x,t,u)) &= ite(b, d(s)(t), d(s)(u)), & (2)\\ d(cons(b,s))(empty) &= empty. & (3)\end{aligned}$$

The conjunction of (2) and (3) is $\mathcal{A}$-equivalent to the equation

$$\begin{aligned} d(cons(b,s)) \;=\; & case\{bjoin.\lambda(lab,t,u).ite(b, d(s)(t), d(s)(u)),\\ & empty.\lambda z.empty\}.\end{aligned}$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ \alpha &= \lambda z.id_{btree},\\ d \circ cons &= (\lambda(b,f).case\{bjoin.\lambda(lab,t,u).ite(b, f(t), f(u)),\\ &\qquad empty.\lambda z.empty\}) \circ (id_2 \times d).\end{aligned}$$

19. Suppose that Σ includes $coBintree(L)$ and $\mathcal{A}|_{coBintree(L)}$ is initial in $Alg_{coBintree(L)}$ and thus isomorphic to $Bin(L)$ (see sample algebra 9.6.12).

Let $d = subtreeC : state \to (btree \to btree)$ denote the function that maps a node $s \in 2^\infty$ of a finite or infinite binary tree t with node labels from L to the subtree of t with root s

Let $b : 2,\ lab : L,\ t, u, u' : btree,\ f : btree \to btree \in V$. d may be specified by the conditional equations

$$d(\alpha(z))(t) \;=\; t, \tag{4}$$
$$d(cons(b,s))(t) \;=\; ite(b, d(s)(u), d(s)(u')) \;\Leftarrow\; split(t) = \iota_1(lab, u, u'), \tag{5}$$
$$d(cons(b,s))(t) \;=\; t \;\Leftarrow\; split(t) = \iota_2(z). \tag{6}$$

The conjunction of (5) and (6) is $\mathcal{A}$-equivalent to the equation

$$d(cons(b,s)) \;=\; \lambda t.[\lambda(lab,u,u').ite(b, d(s)(u), d(s)(u')), \lambda z.t](split(t)).$$

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{aligned} d \circ \alpha &= \lambda z.id_{btree}, \\ d \circ cons &= (\lambda(b,f).\lambda t.[\lambda(lab,u,u').ite(b, f(u), f(u')),\ \lambda z.t](split(t))) \\ &\quad \circ (id_2 \times d). \end{aligned}$$

Further inductively defined functions

20. The Brzozowski equations

Let $C\Sigma = Reg(X)$ (see section 8.2), $z : 1$, $b, c : 2$, $x : X$, $t, u : state$, $pt, pu : state^X \times state \in V$, $B \in \mathcal{P}_+(X)$ and $(\in) : X \times \mathcal{P}(X) \to 2$ and $max, (*) : 2 \times 2 \to 2$ be $C\Sigma$-arrows defined as usually.

Let $d : state \to state^X \times state$ denote the product extension $\langle \delta, id \rangle$ with the equations

$$\begin{array}{rcl} d(t+u) & = & (\lambda x.(\pi_1(d(t))(x) + \pi_1(d(u))(x)),\ t+u), \\ d(t*u) & = & (\lambda x.(\pi_1(d(t))(x) * u + \widehat{\beta(t)} * \pi_1(d(u))(x)),\ t*u) \\ d(star(t)) & = & (\lambda x.\pi_1(d(t))(x),\ star(t)), \\ d(\overline{B}) & = & (\lambda x.\widehat{x \in B},\ \overline{B}), \\ d(\widehat{b}) & = & (\lambda x.\widehat{0}, \widehat{b}) \end{array}$$

(see sample algebra 9.6.23).

An $\mathcal{A}$-equivalent inductive definition of d reads as follows:

$$\begin{array}{rcl} d \circ par & = & \lambda(pt, pu).(\lambda x.(\pi_1(pt)(x) + \pi_1(pu)(x)),\ \pi_2(pt) + \pi_2(pu)) \circ (d \times d) \\ & & : state \times state \to state^X \times state, \end{array}$$

$$\begin{aligned}
d \circ seq &= \lambda(pt, pu).(\lambda x.((\pi_1(pt)(x) * \pi_2(pu)) + \\
&\qquad\qquad\qquad (\widehat{\pi_2(pt)} * \pi_1(pu)(x)),\ \pi_2(pt) * \pi_2(pu)) \circ (d \times d), \\
&\quad : state \times state \rightarrow state^X \times state, \\
d \circ iter &= \lambda pt.(\lambda x.\pi_1(pt)(x),\ star(\pi_2(pt))) \circ d : state \rightarrow state^X \times state, \\
d \circ \underline{\overline{\ \ }} &= \lambda B.(\lambda x.\widehat{x \in B},\ \overline{B}) : \mathcal{P}(X) \rightarrow state^X \times state, \\
d \circ \underline{\widehat{\ \ }} &= \lambda b.(\lambda x.\widehat{0},\ \widehat{b}) : 2 \rightarrow state^X \times state.
\end{aligned}$$

Hence $g[(\pi_1 \circ g(d))/\delta]$ uniquely solves the δ-equations of the sample algebra 9.6.23 and $\pi_1 \circ g(d)$ is a paramorphism (see chapter 14).

Let $d' : state \rightarrow 2$ denote the function β of sample algebra 9.6.23. An inductive definition of d' reads as follows:

$$\begin{aligned}
\beta \circ par &= (\lambda(b, c).max(b, c)) \circ (\beta \times \beta), \\
\beta \circ seq &= (\lambda(b, c).b * c) \circ (\beta \times \beta), \\
\beta \circ iter &= (\lambda b.1) \circ \beta, \\
\beta \circ \underline{\overline{\ \ }} &= \overline{0}, \\
\beta \circ \underline{\widehat{\ \ }} &= id_2.
\end{aligned}$$

21. Balanced binary trees ([53], Example 4.12)

Let X be a set and $C\Sigma = Bintree(X)$ (see section 8.2), $z : 1$, $x : X$, $m, n : \mathbb{N}$, $b, c : 2$, $t, u : btree \in V$ and

$$max : \mathbb{N} \times \mathbb{N} \to \mathbb{N}, \quad (+1) : \mathbb{N} \to \mathbb{N}, \quad (*) : 2 \times 2 \to 2$$

be defined as usually. Let $d : btree \to \mathbb{N} \times 2$ denote the product extension $\langle height, bal \rangle$ 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 9.6.10).

Equations for d read as follows:

$$\begin{aligned} d(empty(z)) &= (0, true), \\ d(bjoin(x, t, u)) &= (\lambda((m, b), (n, c)).(max(m, n) + 1,\ b * c * (m = n)))(d(t), d(u)). \end{aligned}$$

They are $\mathcal{A}$-equivalent to the following inductive definition of d:

$$\begin{aligned} d \circ bjoin &= (\lambda(x, (m, b), (n, c)).(max(m, n) + 1,\ b * c * (m = n))) \circ (id_X \times d \times d) \\ &\quad : X\ btree \times btree \to \mathbb{N} \times 2, \\ d \circ empty &= \lambda z.(0, true) : 1 \to \mathbb{N} \times 2. \end{aligned}$$

Hence $height = \pi_1 \circ g(d)$ and $bal = \pi_2 \circ g(d)$.

22. Flatten a finite tree ([72], Example 4), X be a set and $C\Sigma = Tree(X)$ (see section 8.2). Suppose that Σ also includes $List(X)$.

Let $z:1$, $x:X$, $t:tree$, $ts:trees \in V$ and $conc: state \times state \to state$ be interpreted as in example 12 above. Let $d: tree \to list$, $d': trees \to list \in V$ denote the functions that map a (list of) finite tree(s) to its list of the node labels in depthfirst order (see sample algebra 9.6.13).

Equations for d and d' read as follows:

$$\begin{aligned} d(join(x,ts)) &= cons(x, d'(ts)), \\ d'(nil(z)) &= nil(z), \\ d'(cons(t,ts)) &= conc(d(t), d'(ts)). \end{aligned}$$

They are $\mathcal{A}$-equivalent to the following inductive definition of $\{d, d'\}$:

$$\begin{aligned} d \circ join &= cons \circ (id_X \times d') : X \times trees \to list, \\ d' \circ nil &= nil : 1 \to list, \\ d' \circ cons &= conc \circ (d \times d') : tree \times trees \to list. \end{aligned}$$

16.4 Sample coinductive definitions

Let the assumption of Theorem 16.2 hold true. It tells us that a coinductive definition

$$\bigwedge_{d:s\to e\in D} d\circ c_s = C_e\circ \overline{d}. \tag{1}$$

of $C=\{c_s : e_s\to s\mid s\in S\}$ of polynomial constructors has a unique solution g in $\mathcal{A}$.

Often the functions that satisfy (1) are sum extensions of the functions of E and a further function f. If f is an identity, then E specifies a primitive corecursive function or apomorphism (see chapter 14).

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 $C_e : e[e_s/s\mid s\in S]\to e$.

Coinductively defined functions to natural numbers with infinity

Let $D\Sigma = coNat$ (see section 8.3). For $C=\{c : e\to nat\}$, (1) reads as follows:

$$pred\circ c \;=\; (c+id_1)\circ\overline{pred} : e\to nat+1. \tag{2}$$

Let $z : 1,\ m, m', n, n' : nat \in V$.

1. Successor and zero in final *coNat*-algebras are defined non-recursively:

$$\begin{aligned} succ &= obj\{pred.\iota_1\} : nat \to nat, \\ zero &= obj\{pred.\iota_2\} : 1 \to nat \end{aligned}$$

(see section 15.2).

2. Let $infinity : 1 \to nat$ denote the ordinal number ω with the equation

$$pred(infinity(z)) = \iota_1(infinity(z)).$$

By equation (18), $\iota_1 \circ infinity = (infinity + id_1) \circ \iota_1$, i.e., the injection ι_1 can be pushed inwards. This leads to an $\mathcal{A}$-equivalent coinductive definition of *infinity*:

$$pred \circ infinity = (infinity + id_1) \circ \iota_1.$$

3. Suppose that Σ includes $coList(X)$ (see section 8.3) and $\mathcal{A}|_{coList(X)}$ is final in $Alg_{coList(X)}$. Let $x : X,\ s, s' : state \in V$. Let $length : state \to nat$ denote the function that computes the length of a colist. Its original defining Horn clauses read as follows:

$$\begin{aligned} pred(length(s)) = \iota_1(length(s')) = (length + id_1)(\iota_1(s')) \ &\Leftarrow\ split(s) = \iota_1(x, s'), \\ pred(length(s)) = \iota_2(z) = (length + id_1)(\iota_2(z)) \ &\Leftarrow\ split(s) = \iota_2(z) \end{aligned}$$

Their conjunction is $\mathcal{A}$-equivalent to the equation

$$pred(length(s)) = (length + id_1)[\lambda(x, s').\iota_1(s'), \iota_2](split(s)).$$

An $\mathcal{A}$-equivalent coinductive definition of *length* reads as follows:

$$pred \circ length = (length + id_1) \circ [\lambda(x, s').\iota_1(s'), \iota_2] \circ split.$$

4. ([85], Example 2.6.6) Let $add : nat \times nat \to nat$ denote the addition on A_{nat} with the Horn clauses

$$pred(add(m,n)) = \iota_1(add(m',n)) = (add + id_1)(\iota_1(m',n)) \;\Leftarrow\; pred(m) = \iota_1(m'),$$
$$pred(add(m,n)) = \iota_1(add(m,n')) = (add + id_1)(\iota_1(m,n'))$$
$$\Leftarrow\; pred(m) = \iota_2(z) \wedge pred(n) = \iota_1(n'),$$
$$pred(add(m,n)) = \iota_2(z) = (add + id_1)(\iota_2(z)) \;\Leftarrow\; pred(m) = \iota_2(z) \wedge pred(n) = \iota_2(z).$$

Their conjunction is $\mathcal{A}$-equivalent to the equation

$$pred(add(m,n)) = (add + id_1)(\;[\lambda m'.\iota_1(m',n),$$
$$\lambda z.[\lambda n'.\iota_1(m,n'), \iota_2](pred(n))](pred(m))\;).$$

An $\mathcal{A}$-equivalent coinductive definition of *add* reads as follows:

$$pred \circ add \;=\; (add + id_1) \circ \lambda(m,n).[\lambda m'.\iota_1(m',n),$$
$$\lambda z.[\lambda n'.\iota_1(m,n'), \iota_2](pred(n))](pred(m)).$$

Coinductively defined functions to streams

Let X, Y, Z be sets and $D\Sigma = \mathit{Stream}(X)$. Suppose that

$$\begin{aligned}\mathit{Stream}'(Y) &= \mathit{Stream}(Y)[\mathit{state}'/\mathit{state}, \mathit{head}'/\mathit{head}, \mathit{tail}'/\mathit{tail}],\\ \mathit{Stream}''(Z) &= \mathit{Stream}(Z)[\mathit{state}''/\mathit{state}, \mathit{head}''/\mathit{head}, \mathit{tail}''/\mathit{tail}]\end{aligned}$$

(see section 8.3) and $\mathcal{A}|_{\mathit{Stream}'(Y)}$ and $\mathcal{A}|_{\mathit{Stream}''(Z)}$ are final in $\mathit{Alg}_{\mathit{Stream}'(Y)}$ and $\mathit{Alg}_{\mathit{Stream}''(Z)}$, respectively.

For $C = \{c : e \to \mathit{state}\}$, (1) reads as follows:

$$\begin{aligned}\mathit{head} \circ c &= \overline{\mathit{head}} : e \to X, &(3)\\ \mathit{tail} \circ c &= c \circ \overline{\mathit{tail}} : e \to \mathit{state}. &(4)\end{aligned}$$

Let $z : 1,\ x : X,\ y : Y,\ m, n : \mathbb{N},\ s, s' : \mathit{state} \in V$ and $(+), \mathit{min} : \mathbb{R}^2 \to \mathbb{R},\ (<), (\leq) : \mathbb{R}^2 \to 2$ be defined as usually.

5. ([180], Example 4) Let $\mathit{nats} : \mathbb{N} \to \mathit{state}$ denote the function with the equations

$$\begin{aligned}\mathit{head}(\mathit{nats}(n)) &= n\\ \mathit{tail}(\mathit{nats}(n)) &= \mathit{nats}(n+1).\end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of *nats* reads as follows:

$$\begin{aligned} head \circ nats &= id_{\mathbb{N}}, \\ tail \circ nats &= nats \circ (+1). \end{aligned}$$

6. Let $evens : state \to state$ denote the function that, given a stream s, returns the stream of all elements of s at even positions. Equations for *evens* read as follows:

$$\begin{aligned} head(evens(s)) &= head(s) \\ tail(evens(s)) &= evens(tail(tail(s))). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of *evens* reads as follows:

$$\begin{aligned} head \circ evens &= head, \\ tail \circ evens &= evens \circ tail \circ tail. \end{aligned}$$

7. Let $zip : state \times state \to state$ denote the function with the equations

$$\begin{aligned} head(zip(s, s')) &= head(s) \\ tail(zip(s, s')) &= zip(s', tail(s)). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of *zip* reads as follows:

$$\begin{aligned} head \circ zip &= head \circ \pi_1, \\ tail \circ zip &= zip \circ \langle \pi_2, tail \circ \pi_1 \rangle. \end{aligned}$$

8. Let $blink, blink' : 1 \to state$ denote the constants with the equations

$$\begin{aligned} head(blink) &= 0 \\ tail(blink) &= blink' \\ head(blink') &= 1 \\ tail(blink') &= blink. \end{aligned}$$

Let $c : 1 + 1 \to state$ denote the sum extension $[blink, blink']$ with the equations

$$\begin{aligned} head(c(\iota_1(z))) &= 0, \\ head(c(\iota_2(z))) &= 1, \\ tail(c(\iota_1(z))) &= c(\iota_2(z)), \\ tail(c(\iota_2(z))) &= c(\iota_1(z)). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} head \circ c &= [\overline{0}, \overline{1}], \\ tail \circ c &= c \circ [\iota_2, \iota_1]. \end{aligned}$$

Hence $g[(g(c) \circ \iota_1)/blink][(g(c) \circ \iota_2)/blink']$ uniquely solves the above equations for $\{blink, blink'\}$.

9. Let $X = \mathbb{R}$ and $appzeros : \mathbb{R} \to state$ denote the function that appends its argument to the stream of zeros. Its original defining equations read as follows:

$$\begin{aligned} head(appzeros(x)) &= x, \\ tail(appzeros(x)) &= appzeros(0). \end{aligned}$$

Here is an $\mathcal{A}$-equivalent coinductive definition of *appzeros*:

$$\begin{aligned} head \circ appzeros &= id_{\mathbb{R}}, \\ tail \circ appzeros &= appzeros \circ \overline{0}. \end{aligned}$$

10. ([26], Example 2.5; [157], Theorem 3.1) Let $X = \mathbb{R}$ and $add : state \times state \to state$ denote the addition on A_{state} with the equations

$$\begin{aligned} head(add(s, s')) &= head(s) + head(s'). \\ tail(add(s, s')) &= add(tail(s), tail(s')). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of *add* reads as follows:

$$\begin{aligned} head \circ add &= (+) \circ (head \times head), \\ tail \circ add &= add \circ (tail \times tail). \end{aligned}$$

11. ([180], Example 14 - a *futumorphism*) Let $exchange : state \to state$ denote the function with the equations

$$\begin{aligned} head(exchange(s)) &= head(tail(s)), \\ head(tail(exchange(s))) &= head(s), \\ tail(tail(exchange(s))) &= exchange(tail(tail(s))). \end{aligned}$$

Let $c : state + state \to state$ denote the sum extension $[exchange, tail \circ exchange]$ with the equations

$$\begin{aligned} head(c(\iota_1(s))) &= head(tail(s)), \\ head(c(\iota_2(s))) &= head(s), \\ tail(c(\iota_1(s))) &= c(\iota_2(s)), \\ tail(c(\iota_2(s))) &= c(\iota_1(tail(tail(s)))). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} head \circ c &= [head \circ tail, head], \\ tail \circ c &= c \circ [\iota_2, \iota_1 \circ tail \circ tail]. \end{aligned}$$

Hence $g[(g(c) \circ \iota_1)/exchange]$ uniquely solves the above equations for $exchange$.

12. ([72], Example 10) Let $X = \mathbb{N}$ and $nats\&squares : \mathbb{N} \to state$ denote 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:

$$\begin{aligned} head(nats\&squares(n)) &= n, \\ head(tail(nats\&squares(n))) &= n * n, \\ tail(tail(nats\&squares(n))) &= nats\&squares(n+1). \end{aligned}$$

Let $c : \mathbb{N} + \mathbb{N} \to state$ denote the sum extension $[nats\&squares, tail \circ nats\&squares]$ with the equations

$$\begin{aligned} head(c(\iota_1(n))) &= n, \\ head(c(\iota_2(n))) &= n * n, \\ tail(c(\iota_1(n))) &= c(\iota_2(n))), \\ tail(c(\iota_2(n))) &= c(\iota_1(n+1))). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} head \circ c &= [id_{\mathbb{N}}, \lambda n.n * n], \\ tail \circ c &= c \circ [\iota_2, \iota_1 \circ (+1)]. \end{aligned}$$

Hence $g[(g(c) \circ \iota_1)/nats\&squares]$ uniquely solves the above equations for $nats\&squares$.

13. Let $iterate : X^X \times X \to state$ denote the function with the equations

$$\begin{aligned} head(iterate(f,x)) &= x, \\ tail(iterate(f,x)) &= iterate(f,f(x)). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of *iterate* reads as follows:

$$\begin{aligned} head \circ iterate &= \pi_2, \\ tail \circ iterate &= iterate \circ \lambda(f,x).(f,f(x)). \end{aligned}$$

14. ([37], Example 2.3) Let $s' : state'$, $f : Y^X$, $g : Z^{X\times Y} \in V$ and

$$\begin{aligned} &map : Y^X \times state \to state', \\ &zipWith : Z^{X\times Y} \times state \times state' \to state'' \end{aligned}$$

be variables denoting the functions with the equations

$$\begin{aligned} head'(map(f,s)) &= f(head(s)), \\ tail'(map(f,s)) &= map(f,tail(s)), \\ head''(zipWith(g,s,s')) &= g(head(s),head'(s')), \\ tail''(zipWith(g,s,s')) &= zipWith(g,tail(s),tail(s')). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} head' \circ map &= \lambda(f,s).f(head(s)), \\ tail' \circ map &= map \circ \lambda(f,s).(f, tail(s)), \\ head'' \circ zipWith &= \lambda(g,s,s').g(head(s), head(s')), \\ tail'' \circ zipWith &= zipWith \circ \lambda(g,s,s').(g, tail(s), tail(s')). \end{aligned}$$

15. Let $f : Y^{Y\times X} \in V$ and $foldprefixes : Y^{Y\times X} \times Y \times state \to state'$ denote the function wth the equations

$$\begin{aligned} head(foldprefixes(f,y,s)) &= f(y, head(s)), \\ tail(foldprefixes(f,y,s)) &= foldprefixes(f, f(y, head(s)), tail(s)). \end{aligned}$$

Hence, if $A_{state} = X^{\mathbb{N}}$ and $A_{state'} = Y^{\mathbb{N}}$, then $foldprefixes(f,y,s)$ denotes the function

$$\lambda n.foldl(f,y)(take(n)(s)).$$

An instance of *foldprefixes* is given by [37], Example 2.5.

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} head \circ foldprefixes &= \lambda(f,y,s).f(y, head(s)), \\ tail \circ foldprefixes &= foldprefixes \circ \lambda(f,y,s).(f, f(y, head(s)), tail(s)). \end{aligned}$$

16. Suppose that Σ includes Nat and $\mathcal{A}|_{Nat}$ is initial in Alg_{Nat} and thus isomorphic to $\mathbb{N}$. Define $cycleNats : nat \to state$ such that $cycleNats^{\mathcal{A}}$ is the function that maps $n \in \mathbb{N}$ to the stream of repetitions of the list $[n, n-1, \ldots, 0]$ (see [1], section 2.1). Together with an auxiliary function $c : nat \times nat \to state$, $cycleNats$ may be specified by the equations

$$\begin{aligned} cycleNats(n) &= c(n,n), \\ head(c(m,n)) &= m, \\ tail(c(zero(z),n)) &= c(n,n), \\ tail(c(succ(m),n)) &= c(m,n). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} head \circ c &= \pi_1, \\ tail \circ c &= c \circ \lambda(m,n).case\{zero.\lambda z.(n,n), succ.\lambda m.(m,n)\}(m). \end{aligned}$$

17. ([37], Example 2.2) Let $X = \mathbb{R}$ and $merge : state \times state \to state$ denote the function with the equations

$$head(merge(s,s')) = min(head(s), head(s')),$$

$$\begin{aligned} tail(merge(s,s')) &= ite(head(s) < head(s'), merge(tail(s),s'), \\ &\qquad ite(head(s) = head(s'), \\ &\qquad\quad merge(tail(s),tail(s')), merge(s,tail(s')))). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} head \circ merge &= \lambda(s,s').min(head(s),head(s')), \\ tail \circ merge &= merge \circ \lambda(s,s').ite(head(s) < head(s'), (tail(s),s'), \\ &\qquad ite(head(s) = head(s'), \\ &\qquad\quad (tail(s),tail(s')), (s,tail(s')))). \end{aligned}$$

18. ([180], Example 10) Let $X = \mathbb{R}$ and $insert : \mathbb{R} \times state \to state$ denote the function with the equations

$$\begin{aligned} head(insert(x,s)) &= min(x, head(s)), \\ tail(insert(x,s)) &= ite(x \leq head(s), s, insert(x, tail(s))). \end{aligned}$$

Let $c : (\mathbb{R} \times state) + state \to state$ denote the sum extension $[insert, id]$ with the equations

$$\begin{aligned} head(c(\iota_1(x,s))) &= min(x, head(s)), \\ head(c(\iota_2(s))) &= head(s), \end{aligned}$$

$$\begin{aligned} tail(c(\iota_1(x,s))) &= ite(x \leq head(s), c(\iota_2(s)), c(\iota_1(x, tail(s)))), \\ tail(c(\iota_2(s))) &= c(\iota_2(tail(s))). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} head \circ c &= [\lambda(x,s).min(x, head(s)), head], \\ tail \circ c &= c \circ [\lambda(x,s).ite(x \leq head(s), \iota_2(s), \iota_1(x, tail(s))), \iota_2 \circ tail]. \end{aligned}$$

Then $g[(g(c) \circ \iota_1)/insert]$ uniquely solves the above $insert$-equations and $g(c) \circ \iota_1$ is an apomorphism (see chapter 14).

Coinductively defined functions to colists

Let X, Y, Z 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 section 8.3) and $\mathcal{A}|_{coList'(Y)}$ and $\mathcal{A}|_{coList''(Z)}$ are final in $Alg_{coList'(Y)}$ and $Alg_{coList''(Z)}$, respectively. For $C = \{c : e \to state\}$, (1) reads as follows:

$$split \circ c = (id_X \times c + id_1) \circ \overline{split} : e \to X \times state + 1. \tag{5}$$

Let $z : 1,\ x : X,\ y : Y,\ m, n : \mathbb{N},\ s, s', s_1, s_2 : state \in V$.

19. ([85], Example 2.6.5) The left-append function for colists and the empty colist are defined non-recursively:

$$\begin{aligned} consC &= obj\{split.\iota_1\} : X \times state \to state, \\ nilC &= obj\{split.\iota_2\} : 1 \to state. \end{aligned}$$

20. ([85], Example 2.6.5) Suppose that Σ includes $List'(X) = List(X)[state'/state]$ (see section 8.2) and $\mathcal{A}|_{List'(X)}$ is initial in $Alg_{List'(X)}$. Let $inc : state' \to state \in V$ denote the inclusion of $A_{state'}$ (lists) in A_{state} (lists and colists) with the equations

$$\begin{aligned} split(inc(cons(x,s))) = \iota_1(x, inc(s)) &= \iota_1((id_X \times inc)(x,s)) \\ &= (id_X \times inc + id_1)(\iota_1(x,s)), \\ split(inc(\alpha(z))) = \iota_2(z) = \iota_2(id_1(z)) &= (id_X \times inc + id_1)(\iota_2(z)). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of inc reads as follows:

$$split \circ inc \;=\; (id_X \times inc + id_1) \circ case\{cons.\iota_1, \alpha.\iota_2\}.$$

21. Let $zipWith : Z^{X\times Y} \times state \times state' \to state''$ denote the function that zips two colists with a binary function $f : X \times Y \to Z$ and thus $\mathcal{A}$ satisfies the equations

$$\begin{aligned}
split(zipWith(f,s,s')) = \iota_1(f(x,y), zipWith(s_1,s_2)) \\
= (id_X \times zipWith + id_1)(\iota_1(f(x,y),(s_1,s_2))) \\
\Leftarrow \; split(s) = \iota_1(x,s_1) \wedge split(s') = \iota_1(y,s_2)), \\
split(zipWith(f,s,s')) = \iota_2(z) = (id_X \times zipWith + id_1)(\iota_2(z)) \\
\Leftarrow \; split(s) = \iota_2(z) \vee split(s') = \iota_2(z).
\end{aligned}$$

Their conjunction is $\mathcal{A}$-equivalent to the equation

$$\begin{aligned}
split(zipWith(f,s,s')) \;=\; & (id_X \times zipWith + id_1)([\lambda(x,s_1).[\lambda(y,s_2).\iota_1(f(x,y),(s_1,s_2)), \\
& \iota_2](split(s')), \iota_2](split(s))).
\end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of $zipWith$ reads as follows:

$$\begin{aligned}
split \circ zipWith \;=\; & (id_X \times zipWith + id_1) \circ \\
& \lambda(f,s,s').[\lambda(x,s_1).[\lambda(y,s_2).\iota_1(f(x,y),(s_1,s_2)), \\
& \iota_2](split(s')), \\
& \iota_2](split(s)).
\end{aligned}$$

22. ([85], Example 2.6.5) Let $conc : state \times state \to state$ denote the concatenation of two colists and thus $\mathcal{A}$ satisfies the equations

$$\begin{aligned} split(conc(s,s')) = \iota_1(x, conc(s_1,s')) &= (id_X \times conc + id_1)(\iota_1(x,(s_1,s'))) \\ &\Leftarrow \; split(s) = \iota_1(x,s_1), \\ split(conc(s,s')) = \iota_1(x, conc(s,s_1)) &= (id_X \times conc + id_1)(\iota_1(x,(s,s_1))) \\ &\Leftarrow \; split(s) = \iota_2(z) \wedge split(s') = \iota_1(x,s_1), \\ split(conc(s,s')) = \iota_2(z) &= (id_X \times conc + id_1)(\iota_2(z)) \\ &\Leftarrow \; split(s) = \iota_2(z) \wedge split(s') = \iota_2(z). \end{aligned}$$

Their conjunction is $\mathcal{A}$-equivalent to the equation

$$\begin{aligned} split(conc(s,s')) = (id_X \times conc + id_1)(\; & [\lambda(x,s_1).\iota_1(x, conc(s_1,s)), \\ & \lambda z.[\lambda(x,s_1).\iota_1(x, conc(s,s_1)), \\ & \iota_2](split(s'))](split(s)) \;). \end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of $conc$ reads as follows:

$$\begin{aligned} split \circ conc \;\; = \;\; (id_X \times conc + id_1) \circ \lambda(s,s').[& \lambda(x,s_1).\iota_1(x, conc(s_1,s)), \\ & \lambda z.[\lambda(x,s_1).\iota_1(x, conc(s,s_1)), \\ & \iota_2](split(s'))](split(s)). \end{aligned}$$

23. Flatten a cotree

Suppose that Σ includes $coTree(X)$ (see section 8.3) and $\mathcal{A}|_{coTree(X)}$ is final in $Alg_{coTree(X)}$ and thus isomorpic to $Tree_\infty(X)$ (see sample algebra 9.6.18).

Let $t, u : tree,\ ts, us : trees \in V$ and

$$flatten : tree \to state, \qquad flattenL : trees \to state$$

be variables denoting the functions with the conditional equations

$$\begin{aligned} split(flatten(t)) &= \iota_1(root(t), flattenL(subtrees(t))), \\ split(flattenL(ts)) &= \iota_1(root(u), flattenL(conc(subtrees(u), us)) \\ &\qquad \Leftarrow \ split(ts) = \iota_1(u, us), \\ split(flattenL(ts)) &= \iota_2(z) \ \Leftarrow \ split(ts) = \iota_2(z). \end{aligned}$$

Let $c : tree + trees \to state$ denote the sum extension $[flatten, flattenL]$ with the conditional equations

$$\begin{aligned} split(c(\iota_1(t))) &= \iota_1(root(t), c(\iota_2(subtrees(t)))), \\ split(c(\iota_2(ts))) &= \iota_1(root(u), c(\iota_2(conc(subtrees(u), us)))) \\ &\qquad \Leftarrow \ split(ts) = \iota_1(u, us), \\ split(c(\iota_2(ts))) &= \iota_2(z) \ \Leftarrow \ split(ts) = \iota_2(z). \end{aligned}$$

or, equivalently, the uncondtional equations

$$split \circ c \circ \iota_1 = (id_X \times c + id_1) \circ \lambda t.\iota_1(root(t), c(\iota_2(subtrees(t)))),$$
$$split \circ c \circ \iota_2 = (id_X \times c + id_1) \circ \lambda ts.[\lambda(u, us).\iota_1(root(u), \iota_2(conc(subtrees(u), us))), \iota_2](split(ts)).$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned} split \circ c &= [(id_X \times c + id_1) \circ \lambda t.\iota_1(root(t), c(\iota_2(subtrees(t)))), \\ &\quad (id_X \times c + id_1) \circ \lambda ts.[\lambda(u, us).\iota_1(root(u), \iota_2(conc(subtrees(u), us))), \\ &\quad \iota_2](split(ts))] \\ &= (id_X \times c + id_1) \circ [\lambda t.\iota_1(root(t), c(\iota_2(subtrees(t)))), \\ &\quad \lambda ts.[\lambda(u, us).\iota_1(root(u), \iota_2(conc(subtrees(u), us))). \end{aligned}$$

Then $g[(g(c) \circ \iota_1)/flatten][(g(c) \circ \iota_2)/flattenL]$ uniquely solves the above *flatten*- and *flattenL*-equations.

Coinductively defined functions to infinite binary trees

Let R be a semiring and $D\Sigma = \mathit{infBintree}(R)$. For $C = \{c : e \to \mathit{btree}\}$, (1) reads as follows:

$$\begin{aligned} \mathit{root} \circ c &= \overline{\mathit{root}} : e \to R, & (6)\\ \mathit{left} \circ c &= c \circ \overline{\mathit{left}} : e \to \mathit{btree}, & (7)\\ \mathit{right} \circ c &= c \circ \overline{\mathit{right}} : e \to \mathit{btree}. & (8) \end{aligned}$$

Let $x : R,\ t, t_1, t_2 : \mathit{btree} \in V$ and $(+), (*) : R^2 \to R$ be defined as usually.

24. (see [70], section 4.1) Let $\mathit{mirror} : \mathit{btree} \to \mathit{btree}$ denote the function that mirrors infinite binary trees, i.e., elements of A_{btree}. A coinductive definition of mirror reads as follows:

$$\begin{aligned} \mathit{root} \circ \mathit{mirror} &= \mathit{root},\\ \mathit{left} \circ \mathit{mirror} &= \mathit{mirror} \circ \mathit{right},\\ \mathit{right} \circ \mathit{mirror} &= \mathit{mirror} \circ \mathit{left}. \end{aligned}$$

25. (see [167], sections 4 and 5) Let $appzeros : R \to btree$ denote the function that maps $x \in R$ to the infinite binary tree whose root is labelled with its argument and its other nodes are labelled with 0. A coinductive definition of *appzeros* reads as follows:

$$\begin{aligned} root \circ appzeros &= id_R \\ left \circ appzeros &= appzeros \circ \overline{0}, \\ right \circ appzeros &= appzeros \circ \overline{0}. \end{aligned}$$

Define $add : btree \times btree \to btree$ such that $add^{\mathcal{A}}$ is addition on A_{btree}. A coinductive definition reads as follows:

$$\begin{aligned} root \circ add &= (+) \circ (root \times root), \\ left \circ add &= add \circ (left \times left), \\ right \circ add &= add \circ (right \times right). \end{aligned}$$

Let $pow : R \to btree$ denote the function that maps $x \in R$ to the infinite binary tree whose nodes at level n are labelled with $2^n * x$. A coinductive definition of *pow* reads as follows:

$$\begin{aligned} root \circ pow &= id_R, \\ left \circ pow &= pow \circ \lambda x.2 * x, \\ right \circ pow &= pow \circ \lambda x.2 * x. \end{aligned}$$

Coinductively defined functions to cobintrees

Let X be a set and $D\Sigma = coBintree(X)$ (see section 8.3). For $C = \{c : e \to btree\}$, (1) reads as follows:

$$split \circ c \;=\; (id_X \times c \times c + id_1) \circ \overline{split} : e \to X \times btree \times btree + 1. \qquad (9)$$

Let $z : 1,\ x : X,\ y : Y,\ m, n : \mathbb{N},\ t, t_1, t_2 : btree \in V$ be defined as usually.

26. The root-append function for cobintrees and the empty cobintree are defined non-recursively:

$$\begin{aligned} bjoinC &= obj\{split.\iota_1\} : X \times btree \times btree \to btree, \\ emptyC &= obj\{split.\iota_2\} : 1 \to btree. \end{aligned}$$

27. (see [82], section 5) Let $mirror : btree \to btree$ denote the function that mirrors a cobintree with the conditional equations

$$\begin{aligned} split(mirror(t)) = \iota_1(x, mirror(t_2), mirror(t_1)) \\ = (id_X \times mirror \times mirror + id_1)(\iota_1(x, t_2, t_1)) \\ \Leftarrow\ split(t) = \iota_1(x, t_1, t_2), \end{aligned}$$

$$split(mirror(t)) = \iota_2(z) = (id_X \times mirror \times mirror + id_1)(\iota_2(z)) \ \Leftarrow\ split(t) = \iota_2(z).$$

Their conjunction is $\mathcal{A}$-equivalent to the unconditional equation

$$split(mirror(t)) = (id_X \times mirror \times mirror + id_1)(split(t)).$$

An $\mathcal{A}$-equivalent coinductive definition of $mirror$ reads as follows:

$$split \circ mirror = (id_X \times mirror \times mirror + id_1) \circ split.$$

A proof that the equation $mirror \circ mirror = id$ holds true in final $coBintree(X)$-algebras is given in [138], section 23.

Coinductively defined functions to cotrees

Let X be a set and $D\Sigma = coTree(X)$ (see section 8.3). For

$$C = \{c : e \to tree,\ c' : e' \to trees\},$$

(1) reads as follows:

$$subtrees \circ c = c' \circ \overline{subtrees} : e \to trees, \tag{10}$$

$$root \circ c = \overline{root} : e \to X, \tag{11}$$

$$split \circ c' = (c \times c' + id_1) \circ \overline{split} : e' \to tree \times trees + 1. \tag{12}$$

Let $z : 1,\ x : X,\ t, t', u, u' : tree,\ ts, ts', us, us' : trees \in V$.

28. Let

$$zipWith : X^{X\times X} \times tree \times tree \to tree,$$
$$zipWith' : X^{X\times X} \times trees \times trees \to trees$$

be variables that denote the functions that zip (two colists of) cotrees with a binary function $f : X \times X \to X$ (cf. Example 21 above for the analogous function on colists and possibly different entry sets X, Y and Z) with the conditional equations

$$\begin{aligned}
subtrees(zipWith(f,t,t')) &= zipWith'(f, subtrees(t), subtrees(t')) \\
&= zipWith'(\langle \pi_1, subtrees \circ \pi_2, subtrees \circ \pi_3\rangle(f,t,t')), \\
root(zipWith(f,t,t')) &= f(root(t), root(t')), \\
split(zipWith'(f,ts,ts')) &= \iota_1(zipWith(f,t,t'), zipWith'(f,us,us')) \\
&= (zipWith \times zipWith' + id_1)(\iota_1((f,t,t'),(f,us,us'))) \\
&\qquad \Leftarrow \; split(ts) = \iota_1(t,us) \wedge split(ts') = \iota_1(t',us'), \\
split(zipWith'(f,ts,ts')) &= \iota_2(z) = (zipWith \times zipWith' + id_1)(\iota_2(z)) \\
&\qquad \Leftarrow \; split(ts) = \iota_2(z) \vee split(ts') = \iota_2(z).
\end{aligned}$$

The conjunction of the last two equations is $\mathcal{A}$-equivalent to the unconditional equation

$$\begin{aligned}
split(zipWith'(f,ts,ts')) \;=\; & (zipWith \times zipWith' + id_1) \\
& ([\lambda(t,us).[\lambda(t',us').\iota_1((f,t,t'),(f,us,us')), \\
& \qquad \iota_2](split(ts')), \iota_2](split(ts))).
\end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of $\{zipWith, zipWith'\}$ reads as follows:

$$\begin{array}{rcl} subtrees \circ zipWith &=& zipWith' \circ \langle \pi_1, subtrees \circ \pi_2, subtrees \circ \pi_3 \rangle, \\ root \circ zipWith &=& \lambda(f,t,t').f(root(t), root(t')), \\ split \circ zipWith' &=& (zipWith \times zipWith' + id_1) \circ \\ && \lambda(f,ts,ts').[\lambda(t,us).[\lambda(t',us').\iota_1((f,t,t'),(f,us,us')), \\ && \qquad \qquad \iota_2](split(ts')), \\ && \qquad \iota_2](split(ts))). \end{array}$$

Coinductively defined functions to word languages

Let X be a set and $D\Sigma = Acc(X)$ (see section 8.3).

For $C = \{c : e \to state\}$, (1) reads as follows:

$$\delta \circ c = c \circ \overline{\delta} : e \to state^X, \tag{13}$$

$$\beta \circ c = \overline{\beta} : e \to 2. \tag{14}$$

29. Let $z : 1,\ x : X = \mathbb{Z} \in V$. Let $esum, osum : 1 \to state$ denote the constants with the equations

$$\begin{aligned}
\delta(esum(z)) &= \lambda x.ite(even(x), esum(z), osum(z)),\\
\beta(esum(z)) &= 1,\\
\delta(osum(z)) &= \lambda x.ite(even(x), osum(z), esum(z)),\\
\beta(osum(z)) &= 0.
\end{aligned}$$

Let $c : 1 + 1 \to state$ denote the sum extension $[esum, osum]$ with the equations

$$\begin{aligned}
\delta(c(\iota_1(z))) &= \lambda x.ite(even(x), c(\iota_1(z)), c(\iota_2(z))),\\
\delta(c(\iota_2(z))) &= \lambda x.ite(even(x), c(\iota_2(z)), c(\iota_1(z))),\\
\beta(c(\iota_1(z))) &= 1,\\
\beta(c(\iota_2(z))) &= 0.
\end{aligned}$$

An $\mathcal{A}$-equivalent coinductive definition of c reads as follows:

$$\begin{aligned}
\delta \circ c &= c \circ [\lambda x.ite(even(x), \iota_1, \iota_2), \lambda x.ite(even(x), \iota_2, \iota_1)],\\
\beta \circ c &= c \circ [\overline{1}, \overline{0}].
\end{aligned}$$

Hence $g[(c\circ\iota_1)/esum][(c\circ\iota_2)/osum]$ uniquely solves the above $\{esum, osum\}$-equations.

Let $Q = 1+1$. By Example 9.17, $(\mathcal{B}, \iota_1())$ and $(\mathcal{B}, \iota_2())$ accept all $(x_1, \ldots, x_n) \in \mathbb{Z}^*$ such that $\sum_{i=1}^n x_i$ is even and odd, respectively.

16.5 Sample biinductive definitions

Let the assumption of Theorem 16.3 hold true. It tells us that a biinductive definition

$$\bigwedge_{c:e_c \to s \in C,\ d:s \to e \in D} \forall x_{c,1} \ldots \forall x_{c,n_c}\ d(c(x_c)) = t_{c,d} \tag{1}$$

of a set C of finitary and polynomial constructors has a unique solution g in $\mathcal{A}$.

Biinductively defined functions to streams

Let X be a set and $D\Sigma = \mathit{Stream}(X)$. Then C is a set of *state*-constructors and (1) reads as follows:

$$\bigwedge_{c:e \to s \in C,\ d \in \{head, tail\}} \forall x_{c,1} \ldots \forall x_{c,n_c}\ d(c(x_c)) = t_{c,d}. \tag{2}$$

1. (see [1], section 3.4; [37], Example 2.10)

A biinductive definition of $add : state \times state \to state$ is given by the first two equations of Example 16.4.10.

Let $X = \mathbb{R}$, $z : 1 \in V$ and $fibs : 1 \to state$ denote the stream of Fibonacci numbers with the equations

$$\begin{aligned} head(fibs(z)) &= 0, \\ head(tail(fibs(z))) &= 1, \\ tail(tail(fibs(z))) &= add(fibs(z), tail(fibs(z))). \end{aligned}$$

For turning these equations into a biinductive definition, we introduce a further constructor $tfibs : 1 \to state$ and replace them by the following ones:

$$\begin{aligned} head(fibs(z)) &= 0, && (3) \\ tail(fibs(z)) &= tfibs(z), && (4) \\ head(tfibs(z)) &= 1, && (5) \\ tail(tfibs(z)) &= add(fibs(z), tfibs(z)). && (6) \end{aligned}$$

By (4), the unique solution of (3)-(6) also yields a unique solution of the first three equations.

Since a biinductive definition includes equations for all constructors occuring on right-hand sides, the first two equations of Example 16.4.10 must be added. Hence the complete biinductive definition defines $C = \{add, fibs, tfibs\}$ and reads as follows:

$$\begin{aligned} head(add(s,s')) &= head(s) + head(s'). \\ tail(add(s,s')) &= add(tail(s), tail(s')), \\ head(fibs(z)) &= 0, \\ tail(fibs(z)) &= tfibs(z), \\ head(tfibs(z)) &= 1, \\ tail(tfibs(z)) &= add(fibs(z), tfibs(z)). \end{aligned}$$

The coinductive definition of $exchange : state \to state$ of Example 16.4.11 cannot be written as a biinductive one because $D\Sigma$-arrows with nested destructors are not flat.

2. (see [157, 159]; [26], Example 3.1)

A biinductive definition of $appzeros : \mathbb{R} \to state$ is given by the first two equations of Example 16.4.9.

Let $X = \mathbb{R}$, $s, s' : state \in V$ and $(*) : \mathbb{R}^2 \to \mathbb{R}$ be defined as usually and $\mathit{shuffle} : state \times state \to state$ denote the function with the equations

$$
\begin{aligned}
head(\mathit{shuffle}(s, s')) &= head(s) * head(s'), && (7)\\
tail(\mathit{shuffle}(s, s')) &= add(\mathit{shuffle}(tail(s), s'), \mathit{shuffle}(s, tail(s'))). && (8)
\end{aligned}
$$

Together with the first two equations of Example 16.4.10, (7) and (8) yield a biinductive definition of $C = \{add, \mathit{shuffle}\}$.

Let $conv : state \times state \to state$ denote the function that computes the convolution product of two streams of real numbers and is specified by the equations

$$
\begin{aligned}
head(conv(s, s')) &= head(s) * head(s') && (9)\\
tail(conv(s, s')) &= add(conv(tail(s), s'), conv(appzeros(head(s)), tail(s'))). && (10)
\end{aligned}
$$

Together with the first two equations of Example 16.4.9 and the first two equations of Example 16.4.10, (9) and (10) yield a biinductive definition of $C = \{appzeros, add, conv\}$.

Biinductively defined functions to infinite binary trees

Let R be a semiring with addition $+$ and multiplication $*$ and $D\Sigma = infBintree(R)$ (see section 8.3). Then C is a set of *btree*-constructors and (1) reads as follows:

$$\bigwedge_{c:e\to s\in C,\ d\in\{root,left,right\}} \forall x_{c,1}\dots\forall x_{c,n_c}\ d(c(x_c)) = t_{c,d}. \tag{11}$$

3. (see [167], section 4) Let $x : R,\ t,t' : btree \in V.$ and $conv : btree \times btree \to btree$ denote the convolution product of two binary trees with the equations

$$\begin{aligned} root(conv(t,t')) &= root(t) * root(t'), & (12)\\ left(conv(t,t')) &= add(conv(left(t),t'),conv(appzeros(root(t)),left(t'))), & (13)\\ right(conv(t,t')) &= add(conv(right(t),t'),conv(appzeros(root(t)),right(t'))). & (14) \end{aligned}$$

For obtaining a biinductive definition of $C = \{conv, appzeros, add\}$, we turn the coinductive definitions of $appzeros : R \to btree$ and $add : btree \times btree \to btree$ (see Example 16.4.25) into applicative equations:

$$\begin{aligned} root(appzeros(x)) &= x, & (15)\\ left(appzeros(x)) &= appzeros(0), & (16) \end{aligned}$$

$$\begin{aligned}
right(appzeros(x)) &= appzeros(0), && (17)\\
root(add(t,t')) &= root(t)+root(t'), && (18)\\
left(add(t,t')) &= add(left(t),left(t')), && (19)\\
right(add(t,t')) &= add(right(t),right(t')). && (20)
\end{aligned}$$

(12)-(20) yield a biinductive definition of C.

4. (see [167], section 5) Let $R=\mathbb{N}$, $z:1$, $x:\mathbb{N}\in V$ and $natt:1\to btree$ denote the infinite binary tree whose traversal in breadthfirst order produces the stream of positive natural numbers, and is specified by the equations

$$\begin{aligned}
root(natt(z)) &= 1, && (21)\\
left(natt(z)) &= add(natt(z),pow(1)), && (22)\\
right(natt(z)) &= add(natt(z),pow(2)). && (23)
\end{aligned}$$

For obtaining a biinductive definition of $C=\{natt,pow,add\}$, we turn the coinductive definition of $pow:R\to btree$ (see Example 16.4.25) into applicative equations:

$$root(pow(x)) = x, \qquad (24)$$

$$left(pow(x)) = pow(2 * x), \tag{25}$$
$$right(pow(x)) = pow(2 * x). \tag{26}$$

(18)-(26) yield a biinductive definition of C.

Biinductively defined functions to word languages

Let X be a set and $D\Sigma = Acc(X)$ (see section 8.3). Then C is a set of *state*-constructors and (1) reads as follows:

$$\bigwedge_{c:e\to s\in C,\ d\in\{\delta,\beta\}} \forall x_{c,1}\dots\forall x_{c,n_c}\ d(c(x_c)) = t_{c,d}. \tag{27}$$

5. (see [154], section 10) Let $t, u : state$, $x : X \in V$, $max, (*) : \mathbb{N}^2 \to \mathbb{N}$ be defined as usually and $par, shuffle : state \times state \to state$ denote the parallel composition and the shuffle product, respectively, of two word languages over X, which are specified by the equations

$$\delta(t + u) = (\pi_x(\delta(t)) + \pi_x(\delta(u)))_{x\in X}, \tag{28}$$
$$\beta(t + u) = max(\beta(t), \beta(s')), \tag{29}$$

$$\delta(shuffle(t,u)) = shuffle(\pi_x(\delta(t)),u) + shuffle(t,\pi_x(\delta(u))), \tag{30}$$
$$\beta(shuffle(t,u)) = \beta(t) * \beta(u). \tag{31}$$

(28)-(31) yield a biinductive definition of $C = \{par, shuffle\}$.

6. Let $(S,C) = Reg(X)$ (see section 8.2). Then $S = \{state\}$ and

$$C = \{par, seq, star, \overline{\_}, \widehat{\_}\}.$$

We turn the Brzozowski automaton (see sample algebra 9.6.23) into $Reg(X)$-equations (see section 9.11) that form a biinductive definition of C: ****

Let $t,u : state$, $x : X$, $c : 2 \in V$ and the $Reg(X)$-arrows $(x\in) : \mathcal{P}(X) \to 2$ and $(*) : 2\times 2 \to 2$ be defined as usually.

$$\begin{aligned}
\delta(t*u) &= (\pi_x(\delta(t))*u+\widehat{\beta(t)}*\pi_x(\delta(u)))_{x\in X},\\
\delta(star(t)) &= (\pi_x(\delta(t))*star(t))_{x\in X},\\
\delta(\overline{B}) &= \lambda x.\widehat{x\in B},\\
\delta(\widehat{c}) &= \lambda x.\widehat{0},\\
\beta(t+u) &= max(\beta(t),\beta(u)),\\
\beta(t*u) &= \beta(t)*\beta(u),\\
\beta(star(t)) &= 1,\\
\beta(\overline{B}) &= 0,\\
\beta(\widehat{c}) &= c.
\end{aligned}$$

Since $Beh(X,2)$ (see sample algebra 9.6.24) is final in $Alg_{Acc(X)}$ (see sample final algebra 9.18.12), Theorem 16.3 (13) implies

$$fold^{Beh(X,2)} = unfold^{Bro(X)} : T_{Reg(X)} \to 2^{X^*}. \tag{29}$$

Since the characteristic function $\chi : \mathcal{P}(X^*) \to 2^{X^*}$ is $Reg(X)$-homomorphic,

$$fold^{Beh(X,2)} = \chi \circ fold^{Lang(X)} \tag{30}$$

(see sample algebra 9.6.19). $fold^{Lang(X)}$ represents the **denotational semantics** of regular expressions, $unfold^{Bro(X)}$ is their **operational semantics**.

The latter function yields a correct **parser for regular languages** because by its definition (see sample final algebra 9.18.12), for all $t \in T_{Reg(X)}$ and $w \in X^*$,

$$\begin{aligned} & fold^{Beh(X,2)}(t)(w) = 1 \overset{(29)}{\Leftrightarrow} unfold^{Bro(X)}(t)(w) = 1 \\ \Leftrightarrow \; & if \; w = \epsilon \; then \; \beta^{Bro(X)}(t) \; else \; unfold^{Bro(X)}(\delta^{Bro(X)}(t)(head(w)))(tail(w)). \end{aligned} \tag{31}$$

Since $\beta^{Bro(X)}$ and $\delta^{Bro(X)}$ are inductively defined (see sample algebra 9.6.23 and sample inductive definition 16.3.20), the parser always terminates.

It can be optimized by simplifying the arguments of $\beta^{Bro(X)}$ and $\delta^{Bro(X)}$ before executing their recursive calls. For this purpose equations that hold true in $Beh(X,2)$ like the following ones may be applied:

$$\begin{aligned} t+t &= t \\ \widehat{0}+t &= t \\ t+\widehat{0} &= t \\ \widehat{1}*t &= t \\ t*\widehat{1} &= t \\ \widehat{0}*t &= \widehat{0} \\ t*\widehat{0} &= \widehat{0} \end{aligned}$$

Analogously to Example 13.5—but much faster, the validity of these equations in $Beh(X,2)$ can be shown by algebraic coinduction.

The optimization goes as follows: Given a set E of $C\Sigma$-equations, if applications of E lead from a $C\Sigma$-term t to a $C\Sigma$-term u, this means formally that (t,u) is in the deductive theory $DTh(E)$ of $(C\Sigma,E)$ (see section 19.14).

Since $DTh(E)$ is sound w.r.t. $Alg_{C\Sigma,E}$, in particular, for all pairs of *ground* terms $(t,u)\in DTh(E)$, $fold^{Beh(X,2)}(t)=fold^{Beh(X,2)}(u)$ and thus

$$unfold^{Bro(X)}(t) = fold^{Beh(X,2)}(t) = fold^{Beh(X,2)}(u) = unfold^{Bro(X)}(t). \tag{32}$$

Now suppose that, referring to (31), there is a simpler ground term v than

$$u =_{def} \delta^{Bro(X)}(t)(head(w))$$

such that $(u, v) \in DTh(E)$. The optimized parser that replaces u by v is correct as well:

$$fold^{Beh(X,2)}(t)(w) = 1 \overset{(31)}{\Leftrightarrow} if\ w = \epsilon\ then\ \beta^{Bro(X)}(t)\ else\ unfold^{Bro(X)}(u)(tail(w))$$
$$\overset{(32)}{\Leftrightarrow} if\ w = \epsilon\ then\ \beta^{Bro(X)}(t)\ else\ unfold^{Bro(X)}(v)(tail(w)).$$

16.6 Direct construction of a minimal acceptor of a regular language

For all $t \in T_{Reg(X),state}$, let $L(t)$ be the language of t, i.e., $L(t) = fold^{Lang}(t)$ (see sample algebra 9.6.19). Hence for all $B \in \mathcal{P}_+(X)$ and $t, t' \in T_{Reg(X),state}$,

$$\begin{aligned} &L(\widehat{0}) = \emptyset,\ L(\widehat{1}) = \{\epsilon\}\ L(\overline{B}) = B, \\ &L(t + t') = L(t) \cup L(t'),\ L(t * t')) = L(t) \cdot L(t'),\ L(star(t)) = L(t)^*. \end{aligned} \tag{1}$$

$A = \mathcal{P}(X^*)$ is the carrier of $\mathcal{A} = Pow(X)$ (see sample algebra 9.6.20). $\mathcal{A}$ is final in $Alg_{Acc(X)}$ (see sample final algebra 9.18.10) and thus by Theorem 9.13 (5), for all $L \subseteq X^*$, $(\langle L \rangle, L)$ is a minimal acceptor of L.

By Theorem 9.13 (3), for all $L \subseteq X^*$,

$$\langle L\rangle = img(id_A^\#(L)) = \{id_A^\#(L)(w) \mid w \in (\delta \cdot X)^*\} \subseteq A. \tag{2}$$

For all $w \in (\delta \cdot X)^*$, $w' \in X^*$ is obtained from w by removing all δs. Hence there is an $Acc(X)$-homomorphism h from $\mathcal{A}$ to the product algebra $\mathcal{A}^{X^*}$ such that for all $w \in (\delta \cdot X)^*$, $id_A^\#(L)(w) = h(L)(w')$, and thus by (2),

$$\langle L\rangle = \{id_A^\#(L)(w) \mid w \in (\delta \cdot X)^*\} = \{h(L)(w) \mid w \in X^*\} \tag{3}$$

Moreover, for all $w \in X^*$ and $x \in X$, Theorem 9.13 (2) implies

$$\begin{aligned} &h(L)(\epsilon) = L, \\ &h(L)(wx) = \delta^{\mathcal{A}}(h(L)(w))(x) = \{v \in X^* \mid xv \in h(L)(w)\}. \end{aligned} \tag{4}$$

Theorem 16.5 ([36], Thm. 4.3 (a); [157], Theorem 10.1; [83], Lemma 8; [160], Thm. 189)

For all $t \in T_{Reg(X),state}$, $\langle L(t)\rangle$ is finite.

Proof by induction on t. For all $x \in X$ and $B \in \mathcal{P}_+(X)$,

$$\begin{aligned} &\delta^{\mathcal{A}}(L(\widehat{0}))(x) = \delta^{\mathcal{A}}(\emptyset)(x) = \{w \in X^* \mid xw \in \emptyset\} = \emptyset, \\ &\delta^{\mathcal{A}}(L(\widehat{1}))(x) = \delta^{\mathcal{A}}(\{\epsilon\})(x) = \{w \in X^* \mid xw \in \{\epsilon\}\} = \emptyset, \\ &\delta^{\mathcal{A}}(\overline{B})(x) = \delta^{\mathcal{A}}(B)(x) = \{w \in X^* \mid xw \in B\} = if\ x \in B\ then\ \{\epsilon\}\ else\ \emptyset \end{aligned}$$

and thus

$$|\langle L(\widehat{0})\rangle| = |\langle L(\emptyset)\rangle| = |\{\emptyset\}| = 1 < \omega,$$
$$|\langle L(\widehat{1})\rangle| = |\langle L(\{\epsilon\}\rangle| = |\{\{\epsilon\},\emptyset\}| = 2 < \omega,$$
$$|\langle L(\overline{B})\rangle| = |\langle B\rangle| = |\{B,\{\epsilon\},\emptyset\}| = 3 < \omega.$$

Suppose that for all $L, L' \subseteq X^*$ and $w \in X^*$,

$$h(L \cup L')(w) = h(L)(w) \cup h(L')(w). \tag{5}$$

Then

$$\begin{aligned} &|\langle L \cup L'\rangle| \stackrel{(3)}{=} |\{h(L \cup L')(w) \mid w \in X^*\}| \stackrel{(5)}{=} |\{h(L)(w) \cup h(L')(w) \mid w \in X^*\}| \\ &\leq |\{h(L)(v) \cup h(L')(w) \mid v, w \in X^*\}| \leq |\{(L_1, L_2) \mid L_1 \in \langle L\rangle,\ L_2 \in \langle L'\rangle\}| \\ &= |\langle L\rangle \times \langle L'\rangle| = |\langle L\rangle| * |\langle L'\rangle| \end{aligned} \tag{6}$$

and thus

$$|\langle L(t+t')\rangle| \stackrel{(1)}{=} |\langle L(t) \cup L(t')\rangle| \stackrel{(6)}{\leq} |\langle L(t)\rangle| * |\langle L(t')\rangle| \stackrel{ind.\ hyp.}{<} \omega.$$

Proof of (5) by induction on w.

$$h(L \cup L')(\epsilon) \stackrel{(4)}{=} L \cup L' \stackrel{(4)}{=} h(L)(\epsilon) \cup h(L')(\epsilon),$$
$$h(L \cup L')(wx) \stackrel{(4)}{=} \{v \in X^* \mid xv \in h(L \cup L')(w)\}$$
$$\stackrel{ind.\ hyp.}{=} \{v \in X^* \mid xv \in h(L)(w) \cup h(L')(w)\} \stackrel{(4)}{=} h(L)(wx) \cup h(L')(wx). \quad ❏$$

For all $w \in X^*$, $f(w) =_{def} (\mathit{if}\ \epsilon \in h(L)(w)\ \mathit{then}\ \{\epsilon\}\ \mathit{else}\ \emptyset)$.

Suppose that for all $L, L' \subseteq X^*$ and $w \in X^*$ there are $n \in \mathbb{N}$ and $w_1, \ldots, w_n \in X^*$ such that

$$h(L \cdot L')(w) = h(L)(w) \cdot L' \cup h(L')(w_1) \cup \cdots \cup h(L')(w_n). \tag{7}$$

Then

$$\begin{aligned}
&|\langle L \cdot L' \rangle| \stackrel{(3)}{=} |\{h(L \cdot L')(w) \mid w \in X^*\}| \\
&\stackrel{(7)}{=} |\{h(L)(w) \cdot L' \cup h(L')(w_1) \cup \cdots \cup h(L')(w_n) \mid w \in X^*\}| \\
&\leq |\{(L'', L_1, \ldots, L_n) \mid L'' \in \langle L \rangle,\ L_1, \ldots, L_n \in \langle L' \rangle, n \in \mathbb{N}\}| = |\langle L \rangle \times \mathcal{P}(\langle L' \rangle)| \\
&= |\langle L \rangle| * 2^{|\langle L' \rangle|}
\end{aligned} \tag{8}$$

and thus

$$|\langle L(t * t') \rangle| \stackrel{(1)}{=} |\langle L(t) \cdot L(t') \rangle| \stackrel{(8)}{\leq} |\langle L(t) \rangle| * 2^{|\langle L(t') \rangle|} \stackrel{ind.\ hyp.}{<} \omega.$$

Proof of (7) by induction on w.

$$h(L \cdot L')(\epsilon) \stackrel{(4)}{=} L \cdot L' \stackrel{(4)}{=} h(L)(\epsilon) \cdot L'.$$

For all $w \in X^*$ and $x \in X$,

$$\begin{aligned}
h(L \cdot L')(wx) &\stackrel{(4)}{=} \{v \in X^* \mid xv \in h(L \cdot L')(w)\} \\
&\stackrel{ind.\ hyp.}{=} \{v \in X^* \mid xv \in h(L)(w) \cdot L' \cup h(L')(w_1) \cup \cdots \cup h(L')(w_n)\}
\end{aligned}$$

$$
\begin{aligned}
&= \{v \in X^* \mid xv \in h(L)(w) \cdot L'\} \cup \textstyle\bigcup_{i=1}^n \{v \in X^* \mid xv \in h(L')(w_i)\} \\
&\overset{(4)}{=} \{v \in X^* \mid xv \in h(L)(w) \cdot L'\} \cup \textstyle\bigcup_{i=1}^n h(L')(w_i x) \\
&= \{v \in X^* \mid xv \in h(L)(w)\} \cdot L' \cup f(w) \cdot \{v \in X^* \mid xv \in L'\} \cup \textstyle\bigcup_{i=1}^n h(L')(w_i x) \\
&\overset{(4)}{=} h(L)(wx) \cdot L' \cup f(w) \cdot h(L')(x) \cup \textstyle\bigcup_{i=1}^n h(L')(w_i x). \qquad ❑
\end{aligned}
$$

Suppose that for all $L \subseteq X^*$ and $w \in X^+$ there are $n > 0$ and $w_1, \dots, w_n \in X^*$ such that

$$h(L^*)(w) = h(L)(w_1) \cdot L^* \cup \dots \cup h(L)(w_n) \cdot L^*. \tag{9}$$

Then

$$
\begin{aligned}
|\langle L^* \rangle| &\overset{(3)}{=} |\{h(L^*)(w) \mid w \in X^*\}| = |\{h(L^*)(\epsilon)\} \cup \{h(L^*)(w) \mid w \in X^+\}| \\
&\overset{(4),(9)}{=} |\{L^*\} \cup \{h(L)(w_1) \cdot L^* \cup \dots \cup h(L)(w_n) \cdot L^* \mid w \in X^+\}| \\
&\leq |\{L^*\}| + |\{(L_1, \dots, L_n) \mid L_1, \dots, L_n \in \langle L \rangle, n > 0\}| = 1 + |\mathcal{P}(\langle L \rangle)| - 1 = 2^{|\langle L \rangle|}
\end{aligned} \tag{10}
$$

and thus

$$|\langle L(star(t)) \rangle| \overset{(1)}{=} |\langle L(t)^* \rangle| \overset{(10)}{\leq} 2^{|\langle L(t) \rangle|} \overset{ind.\ hyp.}{<} \omega.$$

Proof of (9) by induction on w. For all $x \in X$,

$$h(L^*)(x) \overset{(4)}{=} \{v \in X^* \mid xv \in L^*\} = \{v \in X^* \mid xv \in L \cdot L^*\} \\ = \{v \in X^* \mid xv \in L\} \cdot L^* \overset{(4)}{=} h(L)(x) \cdot L^*. \tag{11}$$

For all $w \in X^+$ and $x \in X$,

$$\begin{aligned} & h(L^*)(wx) \overset{(4)}{=} \{v \in X^* \mid xv \in h(L^*)(w)\} \\ & \overset{ind.\ hyp.}{=} \{v \in X^* \mid xv \in h(L)(w_1) \cdot L^* \cup \cdots \cup h(L)(w_n) \cdot L^*\} \\ & = \textstyle\bigcup_{i=1}^n (\{v \in X^* \mid xv \in h(L)(w_i)\} \cdot L^* \cup f(w_i) \cdot \{v \in X^* \mid xv \in L^*\}) \\ & = \textstyle\bigcup_{i=1}^n (\{v \in X^* \mid xv \in h(L)(w_i)\} \cdot L^* \cup f(w_i) \cdot h(L^*)(x)) \\ & \overset{(11)}{=} \textstyle\bigcup_{i=1}^n (h(L)(w_i x) \cdot L^* \cup f(w_i) \cdot h(L)(x) \cdot L^*). \end{aligned}$$

❑

Theorem 16.5 tells us that, given a regular expression $t \in Reg(X)$, the minimal acceptor $(\langle L(t)\rangle, L(t))$ of $L(t)$ is finite. It may be constructed stepwise by stepwise compute $Acc(X)$-derivatives of $L(t)$, checking for their equality with previously obtained derivatives and thus building up $\langle L(t)\rangle$. Since $L(t)$ and its derivatives are usually infinite sets, the equality checks becomes tractable only if we turn from $Pow(X)$ to $Bro(X)$ (see sample algebra 9.6.23), stepwise compute $Acc(X)$-derivatives of the regular expression t itself and perform the equality checks by algebraic or fixpoint coinduction (see chapter 13).

By Lemma 13.3 (4), the kernel of $unfold^{Bro(X)}: Bro(X) \to Pow(X)$ is the greatest $Acc(X)$-congruence on $Bro(X)$. Since for all $t \in T_{Reg(X)}$,

$$L(t) = fold^{Lang(X)}(t) = unfold^{Bro(X)}(t),$$

we conclude that the above procedure ends up with a minimal acceptor $(\mathcal{T}, t)$ of $L(t)$ such that $\mathcal{T}$ is a subalgebra of $\langle t \rangle$ and $unfold^{\mathcal{T}}$ is an $Acc(X)$-isomorphism from $\mathcal{T}$ to $\langle L(t) \rangle$.

This construction of a minimal acceptor of $L(t)$ is direct insofar as it avoids the usual detour via a non-deterministic automaton, its determinization and subsequent minimization (see, e.g., [77], chapter 3).

16.7 Guarded CFGs

Let us extend the parser for regular languages given in section 16.5.6 to a parser for context-free languages. As all recursive-descent parsers for CFLs, this parser works only for non-left-recursive CFGs (see section 9.15).

In section 12.3, we have defined the representation of a CFG $G = (S, X, R)$ as a set E_G of $Reg(X)$-equations over S.

If G is non-left-recursive, then G can be transformed into an equivalent **guarded** CFG G' such that for each rule $s \to w$ of G', $w = \epsilon$ or the first element of w belongs to $\mathcal{P}_+(X)$.

Guarded CFGs without non-singleton sets in rules agree with CFGs in *weak Greibach normal form* in the sense of [189].

Let G be guarded. Each equation $s = \overline{w_1} + \cdots + \overline{w_n}$ of E_G satisfies one of the following two cases:

- For all $1 \leq i \leq n$, $\overline{w_i} = \overline{B_i} * \overline{v_i}$ for some $B_i \in \mathcal{P}_+(X)$ and $v_i \in S_X^*$. (1)
- $w_1 = \epsilon$ and for all $2 \leq i \leq n$, $\overline{w_i} = \overline{B_i} * \overline{v_i}$ for some $B_i \in \mathcal{P}_+(X)$ and $v_i \in S_X^*$. (2)

Given the above guarded CFG, a biinductive definition $\overline{E_G}$ of $C = \{c_s : 1 \to state \mid s \in S\}$ on $Acc(X)$ reads as follows:

For all $s \in S$,

$$\delta \circ c_s = \begin{cases} \langle \sum_{i=1}^{n} \widehat{(x \in B_i * \overline{v_i})} \rangle_{x \in X} & \text{in case (1)}, \\ \langle \sum_{i=2}^{n} \widehat{(x \in B_i * \overline{v_i})} \rangle_{x \in X} & \text{in case (2)}, \end{cases} \tag{3}$$

$$\beta \circ s = \begin{cases} \overline{0} & \text{in case (1)}, \\ \overline{1} & \text{in case (2)}. \end{cases} \tag{4}$$

Note that x, B_i and $x \in B_i$ in (3) are elements of X, $\mathcal{P}(X)$ and 2, respectively, and thus $\sum_{i=1}^{n}(\widehat{x \in B_i} * \overline{v_i})$ is a λ-$Reg(X)$-term over S.

Theorem 16.6

Let BRE be the set of equations of sample biinductive definition 16.5.6 and $\mathcal{A}$ be an $(Reg(X) \cup Acc(X))$-algebra with $\mathcal{A}|_{Reg(X)} = Lang(X)$, $\mathcal{A}|_{Acc(X)} = Pow(X)$ and $g \in L(G)^S$ be a solution of E_G in $\mathcal{A}$.

Then $g' = \lambda z.$ satisfies $BRE \cup \overline{E_G}$.

Proof. $Pow(X)$ is final in $Alg_{Acc(X)}$ (see sample final algebra 9.18.10). Hence by Theorem 16.3, $\mathcal{A}$ satisfies the equations of BRE because they form a biinductive definition of the arrows of $Reg(X)$.

Suppose that $\mathcal{A} \models \overline{E_G}$. Let $s \in S$ and $\{w_1, \ldots, w_n\} = \{w \in S_X^* \mid s \to w \in R\}$. Then

$$s^{\mathcal{A}} = (\overline{w_1} + \cdots + \overline{w_n})^{\mathcal{A}} = \bigcup_{i=1}^{n} \overline{w_i}^{\mathcal{A}} \tag{5}$$

and thus for all $x \in X$,

$$\delta^{\mathcal{A}}(s^{\mathcal{A}})(x) = \delta^{\mathcal{A}}(\bigcup_{i=1}^n \overline{w_i}^{\mathcal{A}})(x) = \{w \in X^* \mid \exists\, 1 \leq i \leq n : x \cdot w \in \overline{w_i}^{\mathcal{A}}\}. \tag{6}$$

Let case (1) hold true. Then for all $1 \leq i \leq n$ there are $B_i \in \mathcal{P}_+(X)$ and $v_i \in S_X^*$ such that

$$\begin{aligned} & x \cdot w \in \overline{w_i}^{\mathcal{A}} \;\Leftrightarrow\; x \cdot w \in (\overline{B_i} * \overline{v_i})^{\mathcal{A}} \;\Leftrightarrow\; x \cdot w \in B_i \cdot \overline{v_i}^{\mathcal{A}} \\ & \Leftrightarrow\; x \in B_i \wedge w \in \overline{v_i}^{\mathcal{A}} \;\Leftrightarrow\; \widehat{x \in B_i}^{\mathcal{A}} = \{\epsilon\} \wedge w \in \overline{v_i}^{\mathcal{A}} \;\Leftrightarrow\; \epsilon \in \widehat{x \in B_i}^{\mathcal{A}} \wedge w \in \overline{v_i}^{\mathcal{A}} \\ & \Leftrightarrow\; \epsilon \cdot w \in (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}} \;\Leftrightarrow\; w \in (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}}. \end{aligned} \tag{7}$$

Hence

$$\begin{aligned} & \delta^{\mathcal{A}}(s^{\mathcal{A}})(x) \overset{(6)}{=} \{w \in X^* \mid \exists\, 1 \leq i \leq n : x \cdot w \in \overline{w_i}^{\mathcal{A}}\} \\ & \overset{(7)}{=} \{w \in X^* \mid \exists\, 1 \leq i \leq n : w \in (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}}\} \\ & = \bigcup_{i=1}^n \{w \in X^* \mid w \in (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}}\} = \bigcup_{i=1}^n (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}} = (\textstyle\sum_{i=1}^n (\widehat{x \in B_i} * \overline{v_i}))^{\mathcal{A}} \end{aligned}$$

and thus

$$(\delta \circ s)^{\mathcal{A}} = \langle \sum_{i=1}^n (\widehat{x \in B_i} * \overline{v_i}) \rangle_{x \in X}^{\mathcal{A}},$$

i.e., (3) holds true. Moreover,

$$\beta^{\mathcal{A}}(s^{\mathcal{A}}) \overset{(5)}{=} \beta^{\mathcal{A}}(\bigcup_{i=1}^n \overline{w_i}^{\mathcal{A}}) \overset{(2)}{=} \beta^{\mathcal{A}}(\bigcup_{i=1}^n (\overline{B_i} * \overline{v_i})^{\mathcal{A}}) = \beta^{\mathcal{A}}(\bigcup_{i=1}^n (B_i \cdot \overline{v_i}^{\mathcal{A}})) = 0$$

and thus $(\beta \circ s)^{\mathcal{A}} = \overline{0}$, i.e., (4) holds true.

Let case (2) hold true. Then $w_1 = \epsilon$ and for all $2 \leq i \leq n$ there are $B_i \in \mathcal{P}_+(X)$ and $v_i \in S_X^*$ such that

$$x \cdot w \in \overline{w_i}^{\mathcal{A}} \overset{(7)}{\Leftrightarrow} w \in (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}}. \tag{8}$$

Hence

$$\begin{aligned}
&\delta^{\mathcal{A}}(s^{\mathcal{A}})(x) \overset{(6)}{=} \{w \in X^* \mid \exists\, 1 \leq i \leq n : x \cdot w \in \overline{w_i}^{\mathcal{A}}\} \\
&\overset{w_1=\epsilon}{=} \{w \in X^* \mid \exists\, 2 \leq i \leq n : x \cdot w \in \overline{w_i}^{\mathcal{A}}\} \\
&\overset{(8)}{=} \{w \in X^* \mid \exists\, 2 \leq i \leq n : w \in (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}}\} \\
&= \textstyle\bigcup_{i=2}^n \{w \in X^* \mid w \in (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}}\} = \bigcup_{i=2}^n (\widehat{x \in B_i} * \overline{v_i})^{\mathcal{A}} = (\sum_{2=1}^n (\widehat{x \in B_i} * \overline{v_i}))^{\mathcal{A}}
\end{aligned}$$

and thus

$$(\delta \circ s)^{\mathcal{A}} = \langle \sum_{i=2}^n (\widehat{x \in B_i} * \overline{v_i}) \rangle_{x \in X}^{\mathcal{A}},$$

i.e., (3) holds true. Moreover,

$$\beta^{\mathcal{A}}(s^{\mathcal{A}}) \overset{(5)}{=} \beta^{\mathcal{A}}(\textstyle\bigcup_{i=1}^n \overline{w_i}^{\mathcal{A}}) \overset{(2)}{=} \beta^{\mathcal{A}}(\{\epsilon\} \cup \bigcup_{i=2}^n \overline{w_i}^{\mathcal{A}}) = 1$$

and thus $(\beta \circ s)^{\mathcal{A}} = \overline{1}$, i.e., (4) holds true. ❏

Corollary 16.7

Let $\mathcal{A}, \mathcal{B}$ be $(Reg(X,S) \cup Acc(X))$-algebras with $\mathcal{A}|_{Reg(X,S)} = Lang(X,G)$ (see Theorem 12.5), $\mathcal{B}|_{Reg(X)} = Lang(X)$, $\mathcal{A}|_{Acc(X)} = \mathcal{B}|_{Acc(X)} = Pow(X)$ and $\mathcal{B} \models E_G$.

Then $\mathcal{A} = \mathcal{B}$, i.e., for all $s \in S$, $L(G)_s = s^{\mathcal{B}}$.

Proof. By Theorem 12.5 (i), $\mathcal{A}$ satisfies E_G. Hence by Theorem 16.6, $\mathcal{A}$ and $\mathcal{B}$ satisfy $BRE \cup \overline{E_G}$. Since $BRE \cup \overline{E_G}$ is a biinductive definition of the arrows of $Reg(X,S)$ on $Acc(X)$, Theorem 16.3 implies $\mathcal{A} = \mathcal{B}$. ❑

Example 16.8 Let $G = (S, X, R) = \text{SAB}$ (see Example 9.10) and $\mathcal{B}$ be the $(Reg(X,S) \cup Acc(X))$-algebra with $\mathcal{B}|_{Reg(X)} = Lang(X)$, $\mathcal{B}|_{Acc(X)} = Pow(X)$ and

$$\begin{aligned} C^{\mathcal{B}} &= \{w \in X^* \mid \#a(w) = \#b(w)\}, \\ A^{\mathcal{B}} &= \{w \in X^* \mid \#a(w) = \#b(w) + 1\}, \\ B^{\mathcal{B}} &= \{w \in X^* \mid \#a(w) = \#b(w) - 1\}. \end{aligned}$$

$\mathcal{B}$ satisfies E_G (see Example 12.4).

Proof.

$$
\begin{aligned}
&C^{\mathcal{B}} = \{w \in X^* \mid \#a(w) = \#b(w)\} \\
&= a \cdot \{w \in X^* \mid \#a(w) + 1 = \#b(w)\} \cup b \cdot \{w \in X^* \mid \#a(w) = \#b(w) + 1\} \cup \{\epsilon\} \\
&= a \cdot B^{\mathcal{B}} \cup b \cdot A^{\mathcal{B}} \cup \{\epsilon\} = (\overline{\{a\}} * B + \overline{\{b\}} * A)^{\mathcal{B}} \\
&A^{\mathcal{B}} = \{w \in X^* \mid \#a(w) = \#b(w) + 1\} \\
&= a \cdot \{w \in X^* \mid \#a(w) = \#b(w)\} \cup \ b \cdot \{w \in X^* \mid \#a(w) = \#b(w) + 2\} \\
&= a \cdot \{w \in X^* \mid \#a(w) = \#b(w)\} \\
&\quad \cup \ b \cdot \{w \in X^* \mid \#a(w) = \#b(w) + 1\} \cdot \{w \in X^* \mid \#a(w) = \#b(w) + 1\} \\
&= a \cdot C^{\mathcal{B}} \cup b \cdot A^{\mathcal{B}} \cdot A^{\mathcal{B}} = (\overline{\{a\}} * C + \overline{\{b\}} * A * A)^{\mathcal{B}}, \\
&B^{\mathcal{B}} = \{w \in X^* \mid \#a(w) = \#b(w) - 1\} \\
&= b \cdot \{w \in X^* \mid \#a(w) = \#b(w)\} \cup \ a \cdot \{w \in X^* \mid \#a(w) = \#b(w) - 2\} \\
&= b \cdot \{w \in X^* \mid \#a(w) = \#b(w)\} \\
&\quad \cup \ a \cdot \{w \in X^* \mid \#a(w) = \#b(w) - 1\} \cdot \{w \in X^* \mid \#a(w) = \#b(w) - 1\} \\
&= b \cdot C^{\mathcal{B}} \cup a \cdot B^{\mathcal{B}} \cdot B^{\mathcal{B}} = (\overline{\{b\}} * C + \overline{\{a\}} * B * B)^{\mathcal{B}}.
\end{aligned}
$$

Since SAB is guarded, Corollary 16.7 implies $L(\text{SAB})_s = s^{\mathcal{B}}$ for all $s \in \{S, A, B\}$. ❏

With the help of equations (3) and (4), the parser for regular languages presented in section 16.5 can be extended to a parser for guarded CFGs.

For this purpose, we extend the Brzozowski automaton $Bro(X)$ (see sample algebra 9.6.23) to the $Reg(X,S)$-algebra $Bro(X,G)$ as follows: For all $s\in S$,

$$\begin{aligned} Bro(X,G)_{state} &= T_{Reg(X,S),state}, \\ \delta(s) &= \begin{cases} \lambda x.\sum_{i=1}^{n}(\widehat{x\in B_i} * \overline{v_i}) & \text{in case (1)}, \\ \lambda x.\sum_{i=2}^{n}(\widehat{x\in B_i} * \overline{v_i}) & \text{in case (2)}, \end{cases} \qquad (10) \\ \beta(s) &= \begin{cases} 0 & \text{in case (1)}, \\ 1 & \text{in case (2)}. \end{cases} \qquad (11) \end{aligned}$$

Since $Beh(X,2)$ (see sample algebra 9.6.24) is final in $Alg_{Acc(X)}$ (see sample final algebra 9.18.12), Theorem 16.3 (13) implies

$$fold^{Beh(X,2)} = unfold^{Bro(X,G)} : T_{Reg(X,S)} \to 2^{X^*}. \qquad (12)$$

$unfold^{Bro(X,G)}$ yields a correct **parser for** G because by its definition (see sample final algebra 9.18.12), for all $s\in S$ and $w\in X^*$,

$$\begin{aligned} & fold^{Beh(X,2)}(s)(w) = 1 \overset{(12)}{\Leftrightarrow} unfold^{Bro(X,G)}(s)(w) = 1 \\ \Leftrightarrow\ & \mathit{if}\ w=\epsilon\ \mathit{then}\ \beta^{Bro(X,G)}(s)\ \mathit{else}\ unfold^{Bro(X,G)}(\delta^{Bro(X,G)}(s)(head(w)))(tail(w)). \end{aligned}$$

Since $\beta^{Bro(X)}$ and $\delta^{Bro(X)}$ are inductively defined (see sample algebra 9.6.23, sample inductive definition 16.3.20 and (10/11) above, the parser always terminates.

16.8 Iterative equations I

By Corollary 16.7, the equational representation E_G of a guarded CFG G (see section 12.3) has a unique solution in the final model of an appropriate destructive signature.

We obtain a similar result for sets $E = \{c_i = t_i\}_{i=1}^n$ of Σ-equations (in the sense of section 9.11) with a constant on the left-hand side and right-hand sides $t_i \not\in \{c_1, \dots, c_n\}$. A biinductive definition $\overline{E}$ of $C = \{c_i\}_{i=1}^n$ reads as follows: For all $1 \leq i \leq n$,

17 Iterative equations II

17.1 Algebraic theories

Given a signature $\Sigma = (S, F)$, iterative equations are Σ-equations with a single variable on the left-hand side. They may represent circular Σ-terms or -flowcharts and can be solved in *algebraic* or *sorted theories* [31, 184, 185], which can be regarded as Σ-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_Σ is restricted to projections and product extensions. In the case of $Sum(\Sigma)$, F consists of destructors with sum target and Arr_Σ is restricted to injections and sum extensions. Hence for all $Pow(\Sigma)$-morphisms $f : e \to e'$, e and e' are sorts or product types, while for all $Sum(\Sigma)$-morphisms $f : e \to e'$, e and e' are sorts or sum types.

Accordingly, the algebraic theories Pow_A and Sum_A defined in [184] (Exs. 2.2, 2.3) and [185] (Exs. 2.4.2, 2.4.7) correspond to Σ-algebras $\mathcal{A} : Pow(\Sigma) \to Set$ and $\mathcal{B} : Sum(\Sigma) \to Set$ with carrier A and B, respectively. For all $Pow(\Sigma)$-morphisms f and $Sum(\Sigma)$-morphisms g, $f^\mathcal{A} : A^I \to A^V$ and $g^\mathcal{A} : V \times A \to O \times A$ for sets I, V, O of variables, which actually represent product or sum indices, respectively.

17.2 Term equations

Let $\Sigma = (S, C)$ be a constructive polynomial signature, and V, I be finite S-sorted sets of "internal" and "input variables", respectively. An S-sorted function

$$E : V \to T_\Sigma(I + V)$$

is called a **system of iterative Σ-equations** if $img(E) \cap (I + V) = \emptyset$.

Hence for all $s \in S$ and $x \in V_s$, $E(x) = c(t)$ for some $c : e \to s \in C$ and $t \in T_\Sigma(I+V)_s$.

Let $\mathcal{A}$ be a Σ-algebra with carrier A. An S-sorted function $f : A^I \to A^V$ **solves E in $\mathcal{A}$** if for all $h \in A^I$,

$$[h, f(h)]^* \circ E = f(h),$$

in other words, if f is a fixpoint of the following step function:

$$\begin{array}{rcl} E^{\mathcal{A}} : (A^I \to A^V) & \to & (A^I \to A^V) \\ f & \mapsto & \lambda h.[h, f(h)]^* \circ E \end{array}$$

17.3 The CPO approach for solving term equations

Let $\mathcal{A}$ be ω-continuous. According to the section 15.6, the partial orders, least elements and suprema of A can be lifted to A^V and then to $A^I \to A^V$, i.e., A^V and $A^I \to A^V$ are ω-CPOs.

$E^{\mathcal{A}}$ is ω-continuous. Hence by Theorem 3.4 (1),

$$(E^{\mathcal{A}})^\infty = \bigsqcup_{n<\omega} (E^{\mathcal{A}})^n(\bot) : A^I \to A^V$$

is the least fixpoint of $E^{\mathcal{A}}$ where $\bot : A^I \to A^V$ 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 17.1 For all $g : V \to CT_\Sigma(I)$,

$$[inc_I, g]^* \circ E = g \quad \Rightarrow \quad f_g \text{ solves } E \text{ in } \mathcal{A} \tag{1}$$

where $f_g : A^I \to A^V$ maps $h \in A^I$ to $V \xrightarrow{g} CT_\Sigma(I) \xrightarrow{h^*_\omega} A$ (see section 15.6).

Proof. Suppose that

$$[inc_I, g]^* \circ E = g. \tag{2}$$

Then for all $h \in A^I$,

$$[h, f_g(h)]^* \circ E = [h, h^*_\omega \circ g]^* \circ E = [h^* \circ inc_I, h^*_\omega \circ g]^* \circ E = [h^*_\omega \circ inc_I, h^*_\omega \circ g]^* \circ E$$
$$= (h^*_\omega \circ [inc_I, g])^* \circ E \overset{Lemma\ 15.8}{=} h^*_\omega \circ [inc_I, g]^* \circ E \overset{(2)}{=} h^*_\omega \circ g = f_g(h),$$

i.e., f_g solves E in $\mathcal{A}$. ❑

Theorem 17.2 (generalization of [55], Thm. 5.2; [183], Thm. 6.15; [117], Satz 17)

Let $E : V \to T_\Sigma(I + V)$ be a system of iterative Σ-equations. There is exactly one $g : V \to CT_\Sigma(I)$ with (2).

Proof. Define the step function $E_C : CT^\perp_\Sigma(I)^V \to CT^\perp_\Sigma(I)^V$ as follows:

For all $g : V \to CT^\perp_\Sigma(I)$,

$$E_C(g) = [inc_I, g]^* \circ E.$$

Since E_C is ω-continuous, Theorem 3.4 (1) implies that

$$E_C^\infty = \bigsqcup_{n<\omega} E_C^n(\perp) : V \to CT^\perp_\Sigma(I)$$

is the least fixpoint of E_C where $\perp : V \to CT^\perp_\Sigma(I)$ maps every $x \in V$ to Ω.

Hence it remains to show that every $g : V \to CT_\Sigma(I)$ with (2) agrees with E_C^∞.

So let $B = \bigcup \mathcal{I}$ and $g : V \to CT_\Sigma(I)$ satisfy (2). Since E_C^∞ 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 $n \in \mathbb{N}$,

$$def([inc_I, g]^*(t)) \cap B^n \;\subseteq\; def([inc_I, E_C^{n+1}(\bot)]^*(t)), \tag{4}$$

in particular, for all $x \in V$,

$$def(g(x)) \cap B^n \;\subseteq\; def(E_C^{n+1}(\bot)(x)). \tag{5}$$

(5) implies

$$def(g(x)) \subseteq \bigcup_{n<\omega} def(E_C^n(\bot)(x)) = def(\bigsqcup_{n<\omega} E_C^n(\bot)(x)) = def(E_C^\infty(x))$$

and thus $g \leq E_C^\infty$. Hence by (3), $g = E_C^\infty$.

Proof of (4) by induction on n.

Let $t \in T_\Sigma(I+V)$, $n \in \mathbb{N}$, $h = [inc_I, g]$ and $h_n = [inc_I, E_C^n(\bot)]$.

Case 1: $t = *$. Then $def([inc_I, g]^*(t)) \cap B^0 = 1$, for all $n > 0$, $def([inc_I, g]^*(t)) \cap B^n = \emptyset$, and for all $n \in \mathbb{N}$, $def([inc_I, E_C^{n+1}(\bot)]^*(t)) = 1$. Hence (4) holds true.

Case 2: $t \in V$ and $E(t) = c(u)$ for some $c : e \to s \in C$ and $u \in T_\Sigma(I+V)_e$. By (2),

$$h^*(t) = g(t) = h^*(E(t)) = h^*(c(u)) = c^{\mathcal{A}}(h^*(u)) = c(h^*(u)), \tag{6}$$

$$\begin{aligned} h^*_{n+1}(t) &= E_C^{n+1}(\bot)(t) = E_C(E_C^n(\bot))(t) = h^*_n(E(t)) = h^*_n(c(u)) = c^{\mathcal{A}}(h^*_n(u)) \\ &= c(h^*_n(u)). \end{aligned} \tag{7}$$

Case 2.1: $n = 0$. Then

$$def(h^*(t)) \cap B^n = def(h^*(t)) \cap B^0 = def(h^*(t)) \cap 1 \stackrel{(6)}{=} 1 \stackrel{(7)}{\subseteq} def(h^*_{n+1}(t)).$$

Case 2.2: $n > 0$. Let $w \in def(h^*(t)) \cap B^n$. By (6), $w \in def(c(h^*(u)))$ and thus $w = bv$ for some $b \in B$ and $v \in def(h^*(u)) \cap B^{n-1}$. By induction hypothesis, $v \in def(h^*_n(u))$. Hence

$$w = bv \in def(c(h^*_n(u))) \stackrel{(7)}{=} def(h^*_{n+1}(t)).$$

Therefore, (4) holds true in both subcases.

Case 3: $t \in I$. Then $def(h^*(t)) \cap B^n = def(t) \cap B^n = 1 = def(t) = def(h^*_{n+1}(t))$.

Case 4: $t = c(u)$ for some $c : e \to s \in C$ and $u \in T_\Sigma(I+V)_e$. Then

$$h^*(t) = h^*(c(u)) = c^{\mathcal{A}}(h^*(u)) = c(h^*(u)), \tag{8}$$

$$h^*_{n+1}(t) = h^*_{n+1}(c(u)) = c^{\mathcal{A}}(h^*_{n+1}(u)) = c(h^*_{n+1}(u)) \tag{9}$$

Case 4.1: $n = 0$. Then

$$def(h^*(t)) \cap B^n = def(h^*(t)) \cap B^0 = def(h^*(t)) \cap 1 \overset{(8)}{=} 1 \overset{(9)}{\subseteq} def(h^*_{n+1}(t)).$$

Case 4.2: $n > 0$. Let $w \in def(h^*(t)) \cap B^n$. By (8), $w \in def(c(h^*(u)))$ and thus $w = bv$ for some $b \in B$ and $v \in def(h^*(u)) \cap B^{n-1}$. By induction hypothesis, $v \in def(h^*_n(u))$. Since $h_n \leq h_{n+1}$ and $CT^{\perp}_{\Sigma}(I) \in Poset^S$, $h^*_n \leq h^*_{n+1}$. Hence

$$w = bv \in def(c(h^*_n(u))) \subseteq def(c(h^*_{n+1}(u))) \overset{(9)}{=} def(h^*_{n+1}(t)).$$

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: $t = ()\{i \to t_i \mid i \in I\} \in T_\Sigma(I+V)_e$ 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 $i \in I$,

$$\pi_i(h^*(t)) = h^*(t_i) = \pi_i(()\{i \to h^*(t_i) \mid i \in I\}),$$
$$\pi_i(h^*_{n+1}(t)) = \pi_i(()\{i \to h^*_{n+1}(t_i) \mid i \in I\}).$$

Hence

$$h^*(t) = ()\{i \to h^*(t_i) \mid i \in I\}, \tag{10}$$
$$h^*_{n+1}(t) = ()\{i \to h^*_{n+1}(t_i) \mid i \in I\} \tag{11}$$

Case 6.1: $n = 0$. Then

$$def(h^*(t)) \cap B^n = def(h^*(t)) \cap B^0 = def(h^*(t)) \cap 1 \stackrel{(10)}{=} 1 \stackrel{(11)}{\subseteq} def(h^*_{n+1}(t)).$$

Case 6.2: $n > 0$. Let $w \in def(h^*(t)) \cap B^n$. By (10), $w \in def(()\{i \to h^*(t_i) \mid i \in I\})$ and thus $w = iv$ for some $i \in I$ and $v \in def(h^*(t_i)) \cap B^{n-1}$. By induction hypothesis, $v \in def(h^*_n(t_i))$. Since $h_n \le h_{n+1}$ and $CT^{\perp}_{\Sigma}(I) \in Poset^S$, $h^*_n \le h^*_{n+1}$. Hence

$$w = iv \in def(()\{i \to h^*_n(t_i) \mid i \in I\}) \subseteq def(()\{i \to h^*_{n+1}(t_i) \mid i \in I\}) \stackrel{(11)}{=} def(h^*_{n+1}(t)).$$

Therefore, (4) holds true in both subcases. ❏

Let $E : V \to T_\Sigma(V)$ be a system of iterative Σ-equations without input and $\mathcal{A}$ be a Σ-algebra with carrier A. Then the above step function $E^{\mathcal{A}}$ reduces to:

$$\begin{aligned} E^{\mathcal{A}} : A^V &\to A^V \\ g &\mapsto g^* \circ E, \end{aligned}$$

and $g \in A^V$ solves E in $\mathcal{A}$ iff $g^* \circ E = g$.

Lemma 17.3 Let $E, E' : V \to T_\Sigma(V)$ be systems of iterative Σ-equations and $\mathcal{A}, \mathcal{B}$ be Σ-algebras with carriers A and B, respectively.

(i) For all Σ-homomorphisms $f : \mathcal{A} \to \mathcal{B}$ and $g \in A^V$,

$$f \circ E^{\mathcal{A}}(g) \;=\; E^{\mathcal{B}}(f \circ g).$$

(ii) For all strict and ω-continuous Σ-homomorphisms $f : \mathcal{A} \to \mathcal{B}$,

$$f \circ (E^{\mathcal{A}})^{\infty} \;=\; (E^{\mathcal{B}})^{\infty}.$$

(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).

$$f \circ E^{\mathcal{A}}(g) \overset{Def.\ E^{\mathcal{A}}}{=} f \circ g^* \circ E \overset{Lemma\ 9.9}{=} (f \circ g)^* \circ E \overset{Def.\ E^{\mathcal{B}}}{=} E^{\mathcal{B}}(f \circ g).$$

Proof of (ii). First we show

$$f \circ (E^{\mathcal{A}})^n(\lambda x.\bot_A) \;=\; (E^{\mathcal{B}})^n(\lambda x.\bot_B) \tag{4}$$

for all $n \in \mathbb{N}$ by induction on n. Since f is strict,

$$f \circ (E^{\mathcal{A}})^0(\lambda x.\bot^A) = f \circ (\lambda x.\bot_A) = \lambda x.\bot_B = (E^{\mathcal{B}})^0(\lambda x.\bot_B).$$

If $n > 0$, then by (i),

$$f \circ (E^{\mathcal{A}})^n(\lambda x.\bot^A) = f \circ E^{\mathcal{A}}((E^{\mathcal{A}})^{n-1}(\lambda x.\bot^A)) \overset{(i)}{=} E^{\mathcal{B}}(f \circ (E^{\mathcal{A}})^{n-1}(\lambda x.\bot^A))$$
$$\overset{ind.\ hyp.}{=} E^{\mathcal{B}}((E^{\mathcal{B}})^{n-1}(\lambda x.\bot_B)) = (E^{\mathcal{B}})^n(\lambda x.\bot_B).$$

Hence (4) holds true, and we conclude (ii) as follows:

$$f \circ (E^{\mathcal{A}})^{\infty} = f \circ \bigsqcup_{n \in \mathbb{N}} (E^{\mathcal{A}})^n(\lambda x.\bot_A) \overset{f\ \omega-continuous}{=} \bigsqcup_{n \in \mathbb{N}} (f \circ (E^{\mathcal{A}})^n(\lambda x.\bot_A))$$
$$\overset{(4)}{=} \bigsqcup_{n \in \mathbb{N}} (E^{\mathcal{B}})^n(\lambda x.\bot_B) = (E^{\mathcal{B}})^{\infty}.$$

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. ❏

17.4 The coalgebraic approach for solving term equations

Theorem 17.2 can also be derived from the finality of CT_Σ in $Alg_{co\Sigma}$ (see section 15.4).

For this purpose, $T_\Sigma(V)$ is turned into the $co\Sigma$-algebra $T_{\Sigma,E}$ that is defined as follows:

- For all $s \in S$, $T_{\Sigma,E}(s) = T_\Sigma(V)_s$.
- For all $c : e \to s \in C$ and $t \in T_\Sigma(V)_e$, $d_s^{T_{\Sigma,E}}(c(t)) = c(t) \in \coprod_{c:e\to s} T_\Sigma(V)_e$.
- For all $s \in S$ and $x \in V_s$, $E(x) = c(t)$ implies $d_s^{T_{\Sigma,E}}(x) = c(t)$.

Theorem 17.4

$E^\dagger =_{def} V \stackrel{inc_V}{\rightarrow} T_\Sigma(V) \stackrel{unfold^{T_{\Sigma,E}}}{\rightarrow} CT_\Sigma$ solves E in CT_Σ uniquely. Moreover,

$$unfold^{T_{\Sigma,E}} \circ inc_{T_\Sigma} = fold^{CT_\Sigma} : T_\Sigma \to CT_\Sigma. \tag{1}$$

Since CT_Σ 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 $unfold^{T_{\Sigma,E}} : T_\Sigma(V) \to CT_\Sigma$ is also Σ-homomorphic. Let $c : e \to s \in C$ 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 section 15.4) implies

$$unfold_s^{T_{\Sigma,E}}(c(t)) = c(unfold_e^{T_{\Sigma,E}}(t)). \tag{2}$$

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 Σ-homomorphic and thus by the definition of $E^\dagger$,

$$unfold^{T_{\Sigma,E}} = (E^\dagger)^* \tag{3}$$

because there is only one Σ-homomorphism $h : T_{\Sigma,E} \to CT_\Sigma$ with $h \circ inc_V = E^\dagger$.

Let $x \in V$, $c : e \to s \in C$, $t \in T_\Sigma(V)_e$ 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 $unfold^{T_{\Sigma,E}}$,

$$unfold_s^{T_{\Sigma,E}}(x) = c(unfold_e^{T_{\Sigma,E}}(t)). \tag{4}$$

Hence

$$(E^\dagger)^*_s(E(x)) = (E^\dagger)^*_s(c(t)) \overset{(3)}{=} unfold_s^{T_{\Sigma,E}}(c(t)) \overset{(2)}{=} c(unfold_e^{T_{\Sigma,E}}(t)) \overset{(4)}{=} unfold_s^{T_{\Sigma,E}}(x)$$
$$= E^\dagger(x),$$

i.e., $E^\dagger$ solves E in CT_Σ.

Let $g : V \to CT_\Sigma$ solve E in CT_Σ.
First we show that the Σ-homomorphism $g^* : T_{\Sigma,E} \to CT_\Sigma$ is also $co\Sigma$-homomorphic.

Let $c : e \to s \in C$ and $t \in T_\Sigma(V)_e$. Then

$$d_s^{CT_\Sigma}(g^*_s(c(t))) = d_s^{CT_\Sigma}(c(g^*_e(t))) = c(g^*_e(t)) = g^*_s(c(t)) = g^*_e(d_s^{T_{\Sigma,E}}(c(t)). \tag{5}$$

Let $x \in V$, $c : e \to s \in C$, $t \in T_\Sigma(V)_e$ and $E(x) = c(t)$. Then

$$\begin{aligned} &d_s^{CT_\Sigma}(g^*_s(x)) = d_s^{CT_\Sigma}(g_s(x)) \overset{g\ solves\ E}{=} d_s^{CT_\Sigma}(g^*(E(x))) = d_s^{CT_\Sigma}(g^*_s(c(t))) \\ &\overset{(5)}{=} g^*_e(d_s^{T_{\Sigma,E}}(c(t))) = g^*_e(d_s^{T_{\Sigma,E}}(E(x))) = g^*_e(d_s^{T_{\Sigma,E}}(x)). \end{aligned} \tag{6}$$

By (5) and (6), g^* is $co\Sigma$-homomorphic.

Suppose that $g, h : V \to CT_\Sigma$ solve E in CT_Σ. Since g^* and h^* are $co\Sigma$-homomorphic and thus agree with each other because CT_Σ 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_Σ is a $co\Sigma$-subalgebra of $T_{\Sigma,E}$. Hence the inclusion $inc_{T_\Sigma} : T_\Sigma \to T_\Sigma(V)$ is $co\Sigma$-homomorphic and thus $unfold^{T_{\Sigma,E}} \circ inc_{T_\Sigma} = unfold^{\mathcal{B}}$ because CT_Σ is final in $Alg_{co\Sigma}$. Above we have shown that $unfold^{T_{\Sigma,E}}$ is Σ-homomorphic. Hence $unfold^{\mathcal{B}} = unfold^{T_{\Sigma,E}} \circ inc_{T_\Sigma}$ is also Σ-homomorphic. Therefore, $unfold^{\mathcal{B}} = fold^{CT_\Sigma}$ because T_Σ is initial in Alg_Σ. ❑

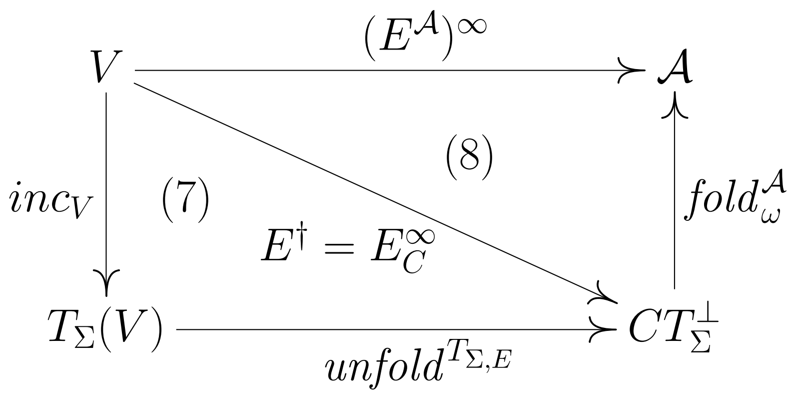
A Σ-term that is representable as a component of the unique solution in CT_Σ of a finite system of iterative Σ-equations is rational (see chapter 2).

The case of empty input

By Theorem 17.2 (with $I = \emptyset$), the equation $g^* \circ E = g$ has exactly one solution $g : V \to CT_\Sigma$, namely E_C^∞. Hence $E^\dagger = E_C^\infty$ and thus triangle (7) in the following diagram commutes.

Let $\mathcal{A}$ be an ω-continuous Σ-algebra with carrier A and $fold_\omega^{\mathcal{A}}$ be defined as in the proof of Theorem 15.7.

Since $fold_\omega^{\mathcal{A}}$ is Σ-homomorphic and E_C^∞ agrees with $(E^{CT_\Sigma^\perp})^\infty$, (8) follows from Lemma 17.3 (ii):



Theorem 17.5 ****

Let the assumptions of Theorem 16.3 hold true, $E : V \to T_{C\Sigma}(V)$ be a system of iterative $C\Sigma$-equations,

$$\Sigma_V = (S, F \cup \{c_x : 1 \to s \mid x \in V_s,\ \ s \in S\}).$$

Theorem 16.3 provides an extension of the final $D\Sigma$-algebra $\mathcal{A}|_{D\Sigma}$ with carrier A to a Σ_V-algebra:

For all $x \in V$, $d : s \to e \in D$ and $g \in A^V$, let $C_{x,d} : e' \to e$ be a $C\Sigma$-arrow, $D_{x,d} : 1 \to e'$ be a flat $D\Sigma$-arrow and $\mathcal{A}_g$ be the Σ-algebra with $\mathcal{A}_g|_\Sigma = \mathcal{A}|_\Sigma$ and $c_x^{\mathcal{A}_g} = g(x)$ for all $x \in V$.

If for all solutions $g \in A^V$ of E in $\mathcal{A}$, $\mathcal{A}_g$ satisfies the biinductive definition

$$\bigwedge_{x\in V,\ d:s\to e\in D} d\circ c_x = C_{x,d}\circ D_{x,d} \tag{1}$$

of V, then E has at most one solution in $\mathcal{A}$.

Proof. Let $g,h \in A^V$ solve E in $\mathcal{A}$. Since $\mathcal{A}|_{D\Sigma}$ is a final $D\Sigma$-algebra, Theorem 16.3 implies that there is a *unique* extension of $\mathcal{A}$ to a Σ_V-algebra that satisfies (1) and Theorem 16.3 (1). By assumption, both $\mathcal{A}_g$ and $\mathcal{A}_h$ satisfy (1). Moreover, $\mathcal{A}$ and thus $\mathcal{A}_g$ and $\mathcal{A}_h$ satisfy Theorem 16.3 (1). Hence $\mathcal{A}_g$ and $\mathcal{A}_h$ agree with each other. In particular, for all $x \in V$, $g(x) = c_x^{\mathcal{A}_g} = c_x^{\mathcal{A}_h} = h(x)$. ❏

The case of nonempty input

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 $A^I \to A^V$.

Let $\Sigma(I)$ be the term grounding of Σ on I (see section 9.12), $\mathcal{A}$ be a Σ-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 $val_s^{\mathcal{A}^g} = g_s$ for all $s \in S$.

For all $g \in CT_\Sigma^I$, the $\Sigma(I)$-homomorphism $g' : CT_{\Sigma(I)} \to CT_\Sigma^g$ is defined as follows:

- For all $s \in S$ and $i \in I_s$, $g'_s(val_s(i)) = g_s(i)$.
- For all $t = x\{i \to t_i \mid i \in I\} \in CT_{\Sigma(I)}$ with $x \notin \{val_s \mid s \in S\}$,
$$g'(t) = x\{i \to g'(t_i) \mid i \in I\}.$$

The S-sorted substitution $\sigma : I + V \to T_{\Sigma(I)}(V)$ assigns x to all $x \in V$ and $val_s(i)$ to all $i \in I_s$, $s \in S$.

Theorem 17.6

Let $\mathcal{A}$ be a Σ-algebra with carrier A.

(1) If $f : A^I \to A^V$ 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
$$[val^{\mathcal{B}}, h]^* \circ E = h.$$

(3) $f : A^I \to A^V$ 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) $$E^\dagger : CT_\Sigma^I \;\to\; CT_\Sigma^V$$
$$g \;\mapsto\; V \stackrel{inc_V}{\to} T_{\Sigma(I)}(V) \stackrel{unfold^{T_{\Sigma(I),\sigma^*\circ E}}}{\longrightarrow} CT_{\Sigma(I)} \stackrel{g'}{\to} CT_\Sigma^g$$

solves E in CT_Σ uniquely.

Since CT_Σ is final in $Alg_{co\Sigma}$, (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 $(f_x : A^I \to A)_{x\in V}$ 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

$$h^* \circ \sigma^* = [val^{\mathcal{B}}, h]^* : T_\Sigma(I+V) \to A \qquad (4)$$

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 $c : e \to s \in C$ and $t \in T_\Sigma(I+V)_e$,

$$h^*(\sigma^*(c(t))) = h^*(c(\sigma^*(t))) = c^{\mathcal{B}}(h^*(\sigma^*(t))) \stackrel{ind.\ hyp.}{=} c^{\mathcal{B}}([val^{\mathcal{B}}, h]^*(t)) = [val^{\mathcal{B}}, h]^*(c(t)).$$

For all sum types $e = \coprod_{i\in J} e_i \in \mathcal{T}_{po}(S)$, $i \in J$ and $t \in T_\Sigma(I+V)_{e_i}$,

$$h^*(\sigma^*(i(t))) = h^*(i(\sigma^*(t))) = \iota_i(h^*(\sigma^*(t))) \stackrel{ind.\ hyp.}{=} \iota_i([val^{\mathcal{B}}, h]^*(t)) = [val^{\mathcal{B}}, h]^*(c(i(t))).$$

For all product types $e = \prod_{i\in J} e_i \in \mathcal{T}_{po}(S)$ and $t = ()\{i \to t_i \mid i \in J\} \in T_\Sigma(I+V)_e$,

$$\pi_i(h^*(\sigma^*(t))) = \pi_i(h^*(\{i \to \sigma^*(t_i) \mid i \in J\})) = h^*(\sigma^*(t_i)) \stackrel{ind.\ hyp.}{=} [val^{\mathcal{B}}, h]^*(t_i)$$
$$= \pi_i(()\{i \to [val^{\mathcal{B}}, h]^*(t_i) \mid i \in J\}) = [val^{\mathcal{B}}, h]^*(t).$$

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 17.2 or 17.4,

$$h = (\sigma^* \circ E)_C^\infty \quad \text{and} \quad h = unfold^{T_{\Sigma(I),\sigma^*\circ E}} \circ inc_V$$

solve $\sigma^* \circ E$ in $CT_{\Sigma(I)}$. Hence by (2), for all $g \in CT_\Sigma^I$,

$$[g, h]^* \circ E = [val^{CT^g_{\Sigma(I)}}, h]^* \circ E = h. \tag{5}$$

Since g' is $\Sigma(I)$-homomorphic,

$$\begin{aligned} g' \circ h &\stackrel{(5)}{=} g' \circ [g, h]^* \circ E \stackrel{Lemma\ 9.9}{=} (g' \circ [g, h])^* \circ E = [g' \circ g, g' \circ h]^* \circ E \\ &\stackrel{img(g) \subseteq CT_\Sigma}{=} [g, g' \circ h]^* \circ E, \end{aligned}$$

i.e., $f : CT_\Sigma^I \to CT_\Sigma^V$ with $f(g) = g' \circ h$ solves E in CT_Σ.

Suppose that $f, f' : CT_\Sigma^I \to CT_\Sigma^V$ solve E in CT_Σ. By (1), for all $g \in CT_\Sigma^I$, $f(g)$ and $f'(g)$ solve $\sigma^* \circ E$ in CT_Σ^g. Hence by Theorem 17.2 or 17.4, $f(g) = f'(g)$. Therefore, E has at most one solution in CT_Σ. ❑

Examples

1. Let $\Sigma = coStream(X)$, $I = \{x, y\}$, $V = \{blink, blink'\}$ and

$$\begin{array}{rcl} E : V & \to & T_\Sigma(I+V) \\ blink & \mapsto & cons(x, blink'), \\ blink' & \mapsto & cons(y, blink). \end{array}$$

The unique solution $g : CT_\Sigma^I \to CT_\Sigma^V$ of E in CT_Σ is defined as follows:

For all $h \in CT_\Sigma^I$ and $w \in \{(),1,2\}^*$,

$$\begin{array}{rcl} g(h)(blink)(w) & = & \begin{cases} cons & \text{if } w \in (()2)^*, \\ h(x) & \text{if } \exists\, n \in \mathbb{N} : w = (()2)^n()1 \wedge even(n), \\ h(y) & \text{if } \exists\, n \in \mathbb{N} : w = (()2)^n()1 \wedge odd(n), \\ \bot & \text{otherwise}, \end{cases} \\ g(h)(blink')(w) & = & g(h)(blink)(()2w). \end{array}$$

Moreover, $cons : X \times state \to state$ and $blink, blink' : 1 \to state$ can be defined biinductively on every final *Stream*-algebra $\mathcal{A}$ with carrier A (e.g., on sample algebra 9.6.5, $InfSeq(X)$) as follows:

$$head \circ cons = \pi_1, \;\; tail \circ cons = \pi_2, \tag{1}$$
$$head \circ blink = 0, \;\; tail \circ blink = blink', \tag{2}$$
$$head \circ blink' = 1, \;\; tail \circ blink' = blink. \tag{3}$$

Let $g \in A^V$ solve E in $\mathcal{A}$, i.e.,

$$g(blink) = cons^{\mathcal{A}}(x, g(blink')),$$
$$g(blink') = cons^{\mathcal{A}}(y, g(blink)).$$

Then $\mathcal{A}_g$ (see Theorem 17.5) satisfies (2) and (3):

$$head^{\mathcal{A}}(g(blink)) = head^{\mathcal{A}}(g(blink)) = head^{\mathcal{A}}(cons^{\mathcal{A}}(x, g(blink'))) \stackrel{(1)}{=} x$$

2. Let $\Sigma = coDAut(\mathbb{Z}, 2)$, $V = \{esum, osum\}$ and

$$\begin{array}{lcl} E : V & \to & T_\Sigma(V) \\ esum & \mapsto & new(()\{\delta \to ()\{x \triangleright even(x) \to esum, x \triangleright odd(x) \to osum \mid x \in \mathbb{Z}\}, \beta \to 1\}) \\ osum & \mapsto & new(()\{\delta \to ()\{x \triangleright even(x) \to osum, x \triangleright odd(x) \to esum \mid x \in \mathbb{Z}\}, \beta \to 0\}). \end{array}$$

The unique solution $g : V \to CT_\Sigma$ of E in CT_Σ is defined as follows:

For all $w \in (\{(),\delta,\beta\} \cup \mathbb{Z})^*$,

$$g(esum)(w) \;=\; \begin{cases} new & \text{if } \exists\, n \in \mathbb{N}, x_1,\dots,x_n \in \mathbb{Z} : w \in (()\delta\mathbb{Z})^*, \\ 1 & \text{if } \exists\, n \in \mathbb{N}, x_1,\dots,x_n \in \mathbb{Z} : \\ & \quad w = ()\delta x_1 \dots ()\delta x_n ()\beta \wedge even(x_1 + \dots + x_n), \\ 0 & \text{if } \exists\, n \in \mathbb{N}, x_1,\dots,x_n \in \mathbb{Z} : \\ & \quad w = ()\delta x_1 \dots ()\delta x_n ()\beta \wedge odd(x_1 + \dots + x_n), \\ \bot & \text{otherwise.} \end{cases}$$

Here δ and β 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_Σ is a $DAut(\mathbb{Z}, 2)$-algebra and $\{g(esum), g(osum)\}$ is the carrier of a $DAut(\mathbb{Z}, 2)$-subalgebra of CT_Σ that is isomorphic to sample algebra 9.6.7.

3. Let $S = \{cmd, exp, bexp\}$, X be a set of program variables that take values in some set Val of values,

$$\begin{aligned} F \;=\; \{ \;& skip : 1 \to cmd,\; assign : X \times exp \to cmd, \\ & seq : cmd \times cmd \to cmd, \\ & cond : bexp \times cmd \times cmd \to cmd \;\}, \end{aligned}$$

$\Sigma = (S, F)$, $I = \{b, c\}$, $V = \{loop\}$ and

$$\begin{aligned} E : V \;&\to\; T_\Sigma(I + V) \\ loop \;&\mapsto\; cond(b, seq(c, loop), skip) \end{aligned}$$

The unique solution $g : CT_\Sigma^I \to CT_\Sigma^V$ of E in CT_Σ is defined as follows:

For all $h \in CT_\Sigma^I$ and $w \in \{(), 1, 2, 3\}^*$,

$$g(h)(loop)(w) \;=\; \begin{cases} cond & \text{if } w \in (()2()2)^*, \\ h(b)(w'') & \text{if } \exists\, w', w'' : w = w'w'' \wedge w' \in (()2()2)^*()1, \\ seq & \text{if } w \in (()2()2)^*()2, \\ skip & \text{if } w \in (()2()2)^*()3, \\ h(c)(w'') & \text{if } \;\exists\, w', w'' : w = w'w'' \wedge \in (((()2()2)^*)()2()1, \\ \bot & \text{otherwise.} \end{cases}$$

Let $\mathcal{A}$ be the ω-continuous Σ-algebra of "store states" that is defined as follows:

Let $St = Val^X$. For all $f : St \to 2$, $g, h : St \to St + 1$, $e : St \to Val$, $x \in X$ and $st \in St$,

$$\begin{aligned}
\mathcal{A}(cmd) &= St \to St + 1, \\
\mathcal{A}(exp) &= St \to Val, \\
\mathcal{A}(bexp) &= St \to 2, \\
skip^{\mathcal{A}} &= id_{St}, \\
assign^{\mathcal{A}}(x, e)(st) &= st[e(st)/x], \\
seq^{\mathcal{A}}(g, h) &= h \circ g, \\
cond^{\mathcal{A}}(f, g, h)(st) &= if\ \ f(st) = 1\ then\ g(st)\ else\ h(st).
\end{aligned}$$

The least solution of E in $\mathcal{A}$ provides the usual semantics of the while-loop operator $while : bexp \times cmd \to cmd$: For all $f : St \to 2$ and $g : St \to St + 1$,

$$while^{\mathcal{A}}(f, g) = (E'^{\mathcal{A}^h})^{\infty}$$

where $h \in A^I$ maps b to f and c to g. ❑

17.5 Flowchart equations

Let $\Sigma = (S, D)$ be a destructive polynomial signature and V, O be S-sorted sets of "internal" and "output variables", respectively. An S-sorted function

$$E : V \to \overline{T_\Sigma}(V + O)$$

is called a **system of iterative Σ-equations** if $img(E) \cap (V + O) = \emptyset$ (see section 9.19).

Hence for all $s \in S$ and $x \in V_s$, $E(x) = d(t)$ for some $d : s \to e \in D$ and $t \in \overline{T_\Sigma}(V + O)_s$.

Let $\mathcal{A}$ be a Σ-algebra with carrier A. A flowchart valuation $g : V \to A_O$ **solves** E **in** $\mathcal{A}$ if

$$[g, \eta_O]^\circ \circ E = g,$$

in other words, if g is the fixpoint of the following step function:

$$\begin{aligned} E^{\mathcal{A}} : A_O^V &\to A_O^V \\ g &\mapsto [g, \eta_O]^\circ \circ E \end{aligned}$$

Since $A_O^V = (V \to A_O) \cong (V \times A) \to (O \times A)$, solutions of flowchart equations are dual to solutions $f : A^I \to A^V$ of term equations (see section 17.2), a dualism that we already know from algebraic theories (see section 17.1).

Let $\mathcal{A}$ be ω-continuous. According to the section 15.6, the partial orders, least elements and suprema of A can be lifted to A_O^V, i.e., A_O^V is an ω-CPO.

$E^{\mathcal{A}}$ is ω-continuous. Hence by Theorem 3.4 (1),

$$(E^{\mathcal{A}})^{\infty} = \bigsqcup_{n<\omega}(E^{\mathcal{A}})^n(\bot) : V \to A_O$$

is the least fixpoint of $E^{\mathcal{A}}$ where for all $s \in S$, $\bot : V \to A_O$ maps the elements of V_s to the least element of $A_{O,s} = (O \times A)^{A_s}$, which maps the elements of A_s to the least element of the sum (!) $O \times A$.

In contrast to Σ-terms, Σ-flowcharts need not form an algebra.

Lemma 17.7 **** For all $g : V \to \overline{CT_\Sigma}(O)$,

$$[g, inc_O]^* \circ E = g \quad \Rightarrow \quad f_g \text{ solves } E \text{ in } \mathcal{A}$$

where $f_g : V \times A \to O \times A$ maps $(x, a) \in V \times A$ to $(\eta_O^+)_\omega(g(x))(a) \in O \times A$ (see section 15.6).

Proof. By definition,

$$curry(f_g) = (\eta_O^+)_\omega \circ g. \tag{1}$$

Suppose that

$$[g, inc_O]^* \circ E = g. \tag{2}$$

Then

$$\begin{aligned}
&[curry(f_g), \eta_O]^+ \circ E \overset{(1)}{=} [(\eta_O^+)_\omega \circ g, \eta_O]^+ \circ E \\
&= [(\eta_O^+)_\omega \circ g, \eta_O^+ \circ inc_O]^+ \circ E \\
&= [(\eta_O^+)_\omega \circ g, (\eta_O^+)_\omega \circ inc_O]^+ \circ E = ((\eta_O^+)_\omega \circ [g, inc_O])^+ \circ E \\
&\overset{Lemma\ 15.9}{=} (\eta_O^+)_\omega \circ [g, inc_O]^* \circ E \overset{(2)}{=} (\eta_O^+)_\omega \circ g \overset{(1)}{=} curry(f_g),
\end{aligned}$$

i.e., f_g solves E in $\mathcal{A}$. ❑

Theorem 17.8 ("dualization" of Theorem SOLC)

Let $E : V \to \overline{T_\Sigma}(V + O)$ be a system of iterative Σ-equations. There is exactly one $g : V \to \overline{CT_\Sigma}(O)$ with (2).

Proof. Define the step function $E_C : \overline{CT_\Sigma^\perp}(O)^V \to \overline{CT_\Sigma^\perp}(O)^V$ as follows:

For all $g : V \to \overline{CT_\Sigma^\perp}(O)$, $E_C(g) = [g, inc_O]^* \circ E$. Since E_C is ω-continuous, Theorem 3.4 (1) implies that

$$E_C^\infty = \bigsqcup_{n<\omega} E_C^n(\perp) : V \to \overline{CT_\Sigma^\perp}(O)$$

is the least fixpoint of E_C where $\perp : V \to \overline{CT_\Sigma^\perp}(O)$ maps every $x \in V$ to Ω.

Hence it remains to show that every $g : V \to \overline{CT_\Sigma}(O)$ with (1) agrees with E_C^∞.

So let $B = \bigcup \mathcal{I}$ and $g : V \to \overline{CT_\Sigma}(O)$ satisfy (1). Since E_C^∞ is the least function that satisfies (1),

$$E_C^\infty \leq g. \tag{3}$$

Below we show that for all $t \in \overline{T_\Sigma}(V + O)$ and $n \in \mathbb{N}$,

$$def([g, inc_O]^*(t)) \cap B^n \ \subseteq \ def([E_C^{n+1}(\perp), inc_O]^*(t)), \tag{4}$$

in particular, for all $x \in V$,

$$def(g(x)) \cap B^n \ \subseteq \ def(E_C^{n+1}(\perp)(x)). \tag{5}$$

(5) implies

$$def(g(x)) \subseteq \bigcup_{n<\omega} def(E_C^n(\perp)(x)) = def(\bigsqcup_{n<\omega} E_C^n(\perp)(x)) = def(E_C^\infty(x))$$

and thus $g \leq E_C^\infty$. Hence by (3), $g = E_C^\infty$.

Proof of (4) by induction on n.

Let $t \in \overline{T_\Sigma}(V+O)$, $n \in \mathbb{N}$, $h = [g, inc_O]$ and $h_n = [E_C^n(\bot), inc_O]$.

Case 1: $t = *$. Then $def([g, inc_O]^*(t)) \cap B^0 = 1$, for all $n > 0$, $def([g, inc_O]^*(t)) \cap B^n = \emptyset$, and for all $n \in \mathbb{N}$, $def([E_C^{n+1}(\bot), inc_O]^*(t)) = 1$. Hence (4) holds true.

Case 2: $t \in V$ and $E(t) = d(u)$ for some $d : s \to e \in D$ and $u \in \overline{T_\Sigma}(V+O)_e$.

By (2),

$$h^*(t) = g(t) = h^*(E(t)) = h^*(d(u)) = d(h^*(u)), \tag{6}$$

$$h_{n+1}^*(t) = E_C^{n+1}(\bot)(t) = E_C(E_C^n(\bot))(t) = h_n^*(E(t)) = h_n^*(d(u)) = d(h_n^*(u)). \tag{7}$$

Case 2.1: $n = 0$. Then

$$def(h^*(t)) \cap B^n = def(h^*(t)) \cap B^0 = def(h^*(t)) \cap 1 \stackrel{(6)}{=} 1 \stackrel{(7)}{\subseteq} def(h_{n+1}^*(t)).$$

Case 2.2: $n > 0$. Let $w \in def(h^*(t)) \cap B^n$. By (6), $w \in def(d(h^*(u)))$ and thus $w = bv$ for some $b \in B$ and $v \in def(h^*(u)) \cap B^{n-1}$. By induction hypothesis, $v \in def(h_n^*(u))$.

Hence

$$w = bv \in def(d(h_n^*(u))) \stackrel{(7)}{=} def(h_{n+1}^*(t)).$$

Therefore, (4) holds true in both subcases.

Case 3: $t \in O$. Then $def(h^*(t)) \cap B^n = def(t) \cap B^n = 1 = def(t) = def(h^*_{n+1}(t))$.

Case 4: $t = d(u)$ for some $d : s \to e \in D$ and $u \in \overline{T_\Sigma}(V+O)_e$. Then

$$h^*(t) = h^*(d(u)) = d(h^*(u)), \tag{8}$$

$$h^*_{n+1}(t) = h^*_{n+1}(d(u)) = d(h^*_{n+1}(u)) \tag{9}$$

Case 4.1: $n = 0$. Then

$$def(h^*(t)) \cap B^n = def(h^*(t)) \cap B^0 = def(h^*(t)) \cap 1 \stackrel{(8)}{=} 1 \stackrel{(9)}{\subseteq} def(h^*_{n+1}(t)).$$

Case 4.2: $n > 0$. Let $w \in def(h^*(t)) \cap B^n$. By (8), $w \in def(c(h^*(u)))$ and thus $w = bv$ for some $b \in B$ and $v \in def(h^*(u)) \cap B^{n-1}$. By induction hypothesis, $v \in def(h^*_n(u))$. Since $h_n \leq h_{n+1}$ and $\overline{CT^{\perp}_\Sigma}(O) \in Poset^S$, $h^*_n \leq h^*_{n+1}$. Hence

$$w = bv \in def(d(h^*_n(u))) \subseteq def(d(h^*_{n+1}(u))) \stackrel{(9)}{=} def(h^*_{n+1}(t)).$$

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: $t = (\,)\{i \to t_i \mid i \in I\} \in \overline{T_\Sigma}(V+O)_e$ for some $e = \coprod_{i\in I} e_i$ and $(t_i)_{i\in I} \in \mathsf{X}_{i\in I} T_\Sigma(V+O)_{e_i}$. Then for all $i \in I$,

$$\pi_i(h^*(t)) = h^*(t_i) = \pi_i((\,)\{i \to h^*(t_i) \mid i \in I\}),$$
$$\pi_i(h^*_{n+1}(t)) = \pi_i((\,)\{i \to h^*_{n+1}(t_i) \mid i \in I\}).$$

Hence

$$h^*(t) = (\,)\{i \to h^*(t_i) \mid i \in I\}, \tag{10}$$
$$h^*_{n+1}(t) = (\,)\{i \to h^*_{n+1}(t_i) \mid i \in I\} \tag{11}$$

Case 6.1: $n = 0$. Then

$$def(h^*(t)) \cap B^n = def(h^*(t)) \cap B^0 = def(h^*(t)) \cap 1 \stackrel{(10)}{=} 1 \stackrel{(11)}{\subseteq} def(h^*_{n+1}(t)).$$

Case 6.2: $n > 0$. Let $w \in def(h^*(t)) \cap B^n$. By (10), $w \in def((\,)\{i \to h^*(t_i) \mid i \in I\})$ and thus $w = iv$ for some $i \in I$ and $v \in def(h^*(t_i)) \cap B^{n-1}$. By induction hypothesis, $v \in def(h^*_n(t_i))$. Since $h_n \leq h_{n+1}$ and $\overline{CT^\perp_\Sigma}(O) \in Poset^S$, $h^*_n \leq h^*_{n+1}$. Hence

$$w = iv \in def((\,)\{i \to h^*_n(t_i) \mid i \in I\}) \subseteq def((\,)\{i \to h^*_{n+1}(t_i) \mid i \in I\}) \stackrel{(11)}{=} def(h^*_{n+1}(t)).$$

Therefore, (4) holds true in both subcases. ❏

17.6 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 (see sample algebra 9.6.3).

$(\mathcal{A}, B)$ **accepts** the **language** $L(\mathcal{A}, B) =_{def} \{w \in X^* \mid fold^{\mathcal{A}}(w) \in B\}$.

$L \subseteq X^*$ 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 $Pow(X)$ (see sample algebra 9.6.20) and the final $NAcc(X)$-algebra $NPow(X)$ (see sample algebra 9.6.21).

Hence the unique $\{N\}Acc(X)$-homomorphism $unfold^{\mathcal{A}} : \mathcal{A} \to \{N\}Pow(X)$ 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.

The Brzozowski automaton (see sample algebra 9.6.23) provides an acceptor for every deterministic regular language:

Since $T_{Reg(X)}$ is a $Acc(X)$-algebra, $Pow(X)$ is a final one, $Lang(X)$ is a $Reg(X)$-algebra (see sample algebra 9.6.19) and

$$fold^{Lang(X)} : T_{Reg(X)} \to Lang(X)$$

is $Acc(X)$-homomorphic,

$$fold^{Lang(X)} = unfold^{Bro(X)}. \tag{1}$$

(1) can also be derived from Theorem 16.3 (12) (see biinductively defined function 16.5.6.

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:

$$\begin{aligned} \delta^{\mathcal{A}'} : \mathcal{P}(A) &\rightarrow \mathcal{P}(A)^X \\ B &\mapsto \lambda x. \textstyle\bigcup_{a \in B} \delta^{\mathcal{A}}(a)(x) \\ \beta^{\mathcal{A}'} : \mathcal{P}(A) &\rightarrow 2 \\ B &\mapsto max\{\beta^{\mathcal{A}}(a) \mid a \in B\} \end{aligned}$$

Suppose that for all $B \subseteq A$,

$$unfold^{\mathcal{A}'}(B) = \bigcup_{a \in B} unfold^{\mathcal{A}}(a). \tag{2}$$

Then in particular, $unfold^{\mathcal{A}'}(\{a\}) = unfold^{\mathcal{A}}(a)$ for all $a \in A$, i.e., $\mathcal{A}'$ and $\mathcal{A}$ accept the same languages. (2) is equivalent to (3): For all $w \in X^*$,

$$w \in unfold^{\mathcal{A}'}(B) \quad \Leftrightarrow \quad \exists\, a \in B : w \in unfold^{\mathcal{A}}(a). \tag{3}$$

Proof of (3) by induction on $|w|$.

$$\epsilon \in unfold^{\mathcal{A}'}(B) \Leftrightarrow max\{\beta^{\mathcal{A}}(a) \mid a \in B\} = \beta^{\mathcal{A}'}(B) = 1 \Leftrightarrow \exists\, a \in B : \beta^{\mathcal{A}}(a) = 1$$
$$\Leftrightarrow \exists\, a \in B : \epsilon \in unfold^{\mathcal{A}}(a).$$

For all $x \in X$ and $w \in X^*$,

$$x \cdot w \in unfold^{\mathcal{A}'}(B) \Leftrightarrow w \in unfold^{\mathcal{A}'}(\delta^{\mathcal{A}'}(B)(x))$$
$$\stackrel{ind.\ hyp.}{\Leftrightarrow} \exists\, b \in \delta^{\mathcal{A}'}(B)(x) = \bigcup_{a \in B} \delta^{\mathcal{A}}(a)(x) : w \in unfold^{\mathcal{A}}(b)$$
$$\Leftrightarrow \exists\, a \in B,\ b \in \delta^{\mathcal{A}}(a)(x) : w \in unfold^{\mathcal{A}}(b) \Leftrightarrow \exists\, a \in B : x \cdot w \in unfold^{\mathcal{A}}(a). \quad ❑$$

(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 $E : A \to T_{Reg(X)}(A)$ such that for all $a \in A$,

$$E(a) = par(\dots(par(seq(x_1, a_1), \; seq(x_2, a_2), \dots), \\ seq(x_n, a_n) \tag{4}$$

or

$$E(a) \; = \; par(\dots(par(\epsilon, seq(x_1, a_1)), \; seq(x_2, a_2)), \dots), \\ seq(x_n, a_n) \tag{5}$$

for some $x_1, \dots, x_n \in X$ and $a_1, \dots, a_n \in A$. (4) and (5) are abbreviated as

$$E(a) \; = \; x_1 \cdot a_1 + x_2 \cdot a_2 + \dots + x_n \cdot a_n \tag{6}$$

and

$$E(a) = 1 + x_1 \cdot a_1 + x_2 \cdot a_2 + \dots + x_n \cdot a_n, \tag{7}$$

respectively.

$g : A \to \mathcal{P}(X^*)$ **solves** E **in** $Lang(X)$ (see sample algebra 9.6.19) if $g^* \circ E = g$.

Let $\mathcal{A}$ be a nondeterministic X-word acceptor with carrier A and

$$E(\mathcal{A}) : A \to T_{Reg(X)}(A)$$

be the system of regular equations that is defined as follows:

For all $a \in A$,

$$E(\mathcal{A})(a) = \begin{cases} \sum\{x \cdot b \mid x \in X,\ b \in \delta^{\mathcal{A}}(a)(x)\} & \text{if } \beta^{\mathcal{A}}(a) = 0 \\ 1 + \sum\{x \cdot b \mid x \in X,\ b \in \delta^{\mathcal{A}}(a)(x)\} & \text{if } \beta^{\mathcal{A}}(a) = 1 \end{cases}$$

$unfold^{\mathcal{A}}$ solves $E(\mathcal{A})$ in $Lang(X)$ uniquely. (8)

Proof. By (1), $h =_{def} unfold^{\mathcal{A}} : A \to \mathcal{P}(X^*)$ is $Reg(X)$-homomorphic.

For the definition of $unfold^{\mathcal{A}}$, see section 9.18.

Let $a \in A$ and case (6) hold true. Then $\beta^{\mathcal{A}}(a) = 0$. Hence

$$h^*(E(\mathcal{A})(a)) = h^*(\sum\{x \cdot b \mid x \in X,\ b \in \delta^{\mathcal{A}}(a)(x)\})$$
$$= \bigcup\{h^*(x) \cdot h^*(b) \mid x \in X,\ b \in \delta^{\mathcal{A}}(a)(x)\} = \bigcup\{x \cdot h(b) \mid x \in X,\ b \in \delta^{\mathcal{A}}(a)(x)\}$$
$$= \{x \cdot w \mid x \in X,\ w \in h(b),\ b \in \delta^{\mathcal{A}}(a)(x)\} = h(a).$$

Let $a \in A$ and case (7) hold true. Then $\beta^{\mathcal{A}}(a) = 1$. Hence

$$h^*(E(\mathcal{A})(a)) = h^*(1 + \sum\{x \cdot b \mid x \in X,\ b \in \delta^{\mathcal{A}}(a)(x)\})$$
$$= h^*(1) \cup \bigcup\{h^*(x) \cdot h^*(b) \mid x \in X,\ b \in \delta^{\mathcal{A}}(a)(x)\}$$
$$= 1 \cup \bigcup\{x \cdot h(b) \mid x \in X,\ b \in \delta^{\mathcal{A}}(a)(x)\}$$
$$= 1 \cup \{x \cdot w \mid x \in X,\ w \in h(b),\ b \in \delta^{\mathcal{A}}(a)(x)\} = h(a).$$

Hence $unfold^{\mathcal{A}}$ solves $E(\mathcal{A})$.

Uniqueness follows from the fact that $E(\mathcal{A})$ is a system of iterative $Reg(X)$-equations (see chapter 17), which allows us to apply Theorem 17.5:

Let $(S,C) = Reg(X)$,

$$C_A = \{a : 1 \to state \mid a \in A\}$$

and for all $a \in A$, $op_{a,\delta} : 1 \to state^X$ and $op_{a,\delta} : 1 \to 2$ 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

$$op_{a,\delta} \;=\; if **** $$

Conversely, let $E : A \to T_{Reg(X)}(A)$ 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:

$$\begin{aligned}
\delta^{\mathcal{A}(E)} : A &\to \mathcal{P}_\omega(A)^X \\
a &\mapsto \lambda x.\{a_i \mid 1 \le i \le n,\ x_i = x\} \text{ if (6) or (7) holds true} \\
\beta^{\mathcal{A}(E)} : A &\to 2 \\
a &\mapsto \begin{cases} 0 & \text{if (6) holds true} \\ 1 & \text{if (7) holds true} \end{cases}
\end{aligned}$$

$unfold^{\mathcal{A}(E)}$ solves E in $Lang(X)$ uniquely. (9)

Proof. Obviously, $E(\mathcal{A}(E)) = E$. Hence (8) implies (9). ❑

17.7 Tree acceptors

The literature on tree acceptors spans over many decades (see, e.g., [179, 152, 32, 176, 149, 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 [176], section 3.2.2.

The tree language to be accepted is a set of Σ-terms where $\Sigma = (S, F)$ is a finitary signature (see chapter 8).

A **bottom-up Σ-term acceptor** is a pair $(\mathcal{A}, B)$ that consists of a Σ-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_Σ is the carrier of an initial Σ-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** Σ**-term acceptor** is a pair $(\mathcal{A}, a)$ that consists of a $TAcc(\Sigma)$-algebra ($NTAcc(\Sigma)$-algebra) $\mathcal{A}$ and an element of the carrier of $\mathcal{A}$ that is called an **initial state**.

$\mathcal{P}(T_\Sigma)$ is the carrier of both the final $TAcc(\Sigma)$-algebra $TPow(\Sigma)$ (sample algebra 9.6.29) and the final $NTAcc(\Sigma)$-algebra $NTPow(\Sigma)$ (sample algebra 9.6.30). Hence the unique $\{N\}TAcc(\Sigma)$-homomorphism $unfold^{\mathcal{A}} : \mathcal{A} \to \{N\}TPow(\Sigma)$ maps states to subsets of T_Σ.

$(\mathcal{A}, a)$ **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, $cl(L)$, is the least subset T of T_Σ that contains L and satifies the following implication: for all $c : s_1 \times \cdots \times s_n \to s \in C$ and $1 \leq i < j \leq n$,

$$c(t_1,\dots,t_{i-1},u_i,t_{i+1},\dots,t_n), c(t_1,\dots,t_{j-1},u_j,t_{j+1},\dots,t_n) \in T$$
$$\Rightarrow \quad c(t_1,\dots,t_{i-1},u_i,t_{i+1},\dots,t_{j-1},u_j,t_{j+1},\dots,t_n) \in T.$$

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 $L = cl(L)$.

$L \subseteq T_\Sigma$ 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 $unfold^\mathcal{A}(a) = L$ 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 $unfold^\mathcal{A}$ (see above), there are $a_1,\dots,a_n$ in the carrier of $\mathcal{A}$ such that $\delta_c(a) = (a_1,\dots,a_n)$, $u_i \in unfold^\mathcal{A}(a_i)$, $u_j \in unfold^\mathcal{A}(a_i)$ and for all $1 \leq i \leq n$, $t_i \in unfold^\mathcal{A}(a_i)$. Hence $c(t_1,\dots,t_{i-1},u_i,t_{i+1},\dots,t_{j-1},u_j,t_{j+1},\dots,t_n) \in unfold^\mathcal{A}(a) = L$. 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 $B \subseteq A$,

$$L(\mathcal{A}', B) = \bigcup_{a \in B} cl(L(\mathcal{A}, a)), \qquad (1)$$

or, equivalently, for all $t \in T_\Sigma$,

$$t \in unfold^{\mathcal{A}'}(B) \quad \Leftrightarrow \quad \exists\, a \in B : t \in cl(unfold^{\mathcal{A}}(a)). \qquad (2)$$

The operations of $\mathcal{A}'$ are defined as follows: For all $c : s_1 \times \cdots \times s_n \to s \in C$,

$$\begin{aligned} \delta_c^{\mathcal{A}'} : \mathcal{P}(A) &\to \mathcal{P}(A)^n \\ B &\mapsto \textstyle\bigcup_{a \in B} \delta_c^{\mathcal{A}}(a) \end{aligned}$$

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 $c : A_1 \times \cdots \times A_n \to s$, $A_1, \dots, A_n \in \mathcal{I}$ and $a_i \in A_i$ for all $1 \le i \le n$. 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

$$t \in unfold^{\mathcal{A}'}(B) \;\Leftrightarrow\; \forall\, 1 \le i \le n : a_i \in unfold^{\mathcal{A}'}(B_i) = B_i.$$

[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, $paths(t)$, 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 $p \in paths(\lambda w.t(iw))$, then $(c,i) : p \in paths(t)$,
- if $t(\epsilon) = c'$, then $paths(t) = \{c'\}$.

Given $L \subseteq T_{C\Sigma}$,

$$\begin{aligned} paths(L) &=_{def} \bigcup\{paths(t) \mid t \in L\}, \\ pclosure(L) &=_{def} \{t \in T_{C\Sigma} \mid paths(t) \subseteq paths(L)\} \end{aligned}$$

are called the **path language** and **path closure** of L, respectively. L is **path closed** if $L = pclosure(L)$.

(1) If $L \subseteq T_{C\Sigma}$ is top-down regular, then $paths(L)$ is regular. (2) If $L \subseteq T_{C\Sigma}$ is top-down regular, then $pclosure(L)$ 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 $C = \{f : state^2 \to state,\ c, d : 1 \to state\}$. The language

$$L \ =_{def}\ \{f(c,d), f(d,c)\} \ \subseteq\ T_{C\Sigma}$$

(mentioned in [42], Prop. 1.6.1, [176], Remark 3.35, and [46], Thm. 10) is not path closed because $f(c,c) \notin L$, although

$$paths(f(c,c)) \ =\ \{[(f,1),c],[(f,2),c]\} \ \subseteq\ \{[(f,1),c],[(f,2),d],[(f,1),d],[(f,2),c]\}$$
$$=\ paths(L).$$

It is also easy to see that for every initial automaton $(\mathcal{A}, a)$ with carrier A is a $TAcc(C)$-algebra,

$$L \subseteq unfold^{\mathcal{A}}(a) \quad \text{implies} \quad f(c,c) \in unfold^{\mathcal{A}}(a):$$

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 $f(c,c) \in unfold^{\mathcal{A}}(a)$. ❏

Path language for trees with infinite paths: Given $t \in CT_{C\Sigma}$, the **path language** of t, $paths(t)$, 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 $p \in X^*$,

- $(c,i) : p \in paths(t)$ implies $t(\epsilon) = c$, $t(i) \in src(t)$ and $p \in paths(\lambda w.t(iw))$,
- $c' : p \in paths(t)$ implies $def(t) = 1$, $t(\epsilon) = c'$ and $p = \epsilon$.

Let $L \subseteq T_\Sigma$, $\sim_L$ be the **Nerode relation of** L, i.e., the greatest Σ-congruence that is contained in the kernel of $\chi(L) : T_\Sigma \to 2$, $Q = \{[t]_{\sim_L} \mid t \in L\}$ and $\mathcal{F}(L) =_{def} (T_\Sigma/{\sim_L}, Q)$. Hence

$$L(\mathcal{F}(L)) = \{t \in T_\Sigma \mid fold^{T_\Sigma/\sim_L}(t) \in Q\} = \{t \in T_\Sigma \mid nat_{\sim_L}(t) \in Q\}$$
$$= \{t \in T_\Sigma \mid t \in L\} = L$$

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.

Top-down tree automata in Kleisli categories

Let $\Sigma = (S, F)$ be a signature. A **nondeterministic Σ-algebra $\mathcal{A}$** consists of an S-sorted set A and a function $f^{\mathcal{A}} : A_e \to \mathcal{P}(A_{e'})$ for every $f : e \to e' \in F$.

Let $\mathcal{A}, \mathcal{B}$ be nondeterministic Σ-algebra with carriers A and B, respectively. A multivalued S-sorted function $h : A \to B$ is a **multivalued Σ-homomorphism** from $\mathcal{A}$ to $\mathcal{B}$ if for all $f : e \to e' \in F$,

$$h_{e'} \circ_{\mathcal{P}} f^{\mathcal{A}} = f^{\mathcal{B}} \circ_{\mathcal{P}} h_e$$

where $\circ_{\mathcal{P}}$ is the Kleisli composition of the powerset monad $\mathcal{P}$ (see chapter 24) and h_e and $h_{e'}$ are instances of a lifting of h to a *multivalued* $\mathcal{T}_{po}(S)$-sorted function.

$ndAlg_\Sigma$ denotes the subcategory of Mfn^S (see chapter 7) that consists of all nondeterministic Σ-algebras and multivalued Σ-homomorphisms.

Let $\Sigma = (S, C)$ be a constructive signature, $(S', D) = co\Sigma$ (see chapter 15). A nondeterministic $co\Sigma$-algebra $\mathcal{L}(\Sigma)$ of tree languages may be defined as follows:

For all $s \in S'$,

$$\begin{array}{rcl}\mathcal{L}(\Sigma)_s & = & T_{\Sigma,s}, \\ d_s^{\mathcal{L}(\Sigma)} : T_{\Sigma,s} & \to & \mathcal{P}(\coprod_{c:e\to s\in C} T_{\Sigma,e}) \\ c\{i \to t_i \mid i \in I\} & \mapsto & \{((t_i)_{i\in I}, c)\}.\end{array}$$

Provided that results of [84], section 5; [83], section 3.2; [68], section 3, can be adopted here, there is a distributive law

$$(\coprod_{c:e\to s\in C} \_e) \circ \mathcal{P} \;\to\; \mathcal{P} \circ \coprod_{c:e\to s\in C} \_e$$

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 $s \in S'$,

$$\begin{array}{rcl}unfold_s^{\mathcal{A}} : A_s & \to & \mathcal{P}(T_{\Sigma,s}) \\ a & \mapsto & (\bigsqcup_{n\in\mathbb{N}} \Phi^n(\lambda x.\emptyset))(a,s) \qquad \text{(see chapter 3)} \\ \Phi : \mathcal{P}(T_\Sigma)^{\coprod_{s\in S'} A_s} & \to & \mathcal{P}(T_\Sigma)^{\coprod_{s\in S'} A_s} \\ f & \mapsto & \lambda(a,s).f(a,s) \cup \{c\{i \to t_i \mid i \in I\} \\ & & \quad \mid (b, c : \prod_{i\in I} s_i \to s) \in d_s^{\mathcal{A}}(a), \\ & & \quad\; t_i \in f(\pi_i(b), s_i) \vee (s_i \in Set_{\neq\emptyset} \wedge t_i \in s_i)\}.\end{array}$$

Initial models of constructive polynomial signatures

Let $\Sigma = (S, F)$ be a constructive polynomial signature, κ be the cardinality of the greatest exponent occurring in the source of some $f \in F$ and λ be the first regular cardinal number $> \kappa$.

By Theorem 14.6 (2), H_Σ is λ-cocontinuous and thus by Theorem 14.7, Alg_{H_Σ} has an initial object $\alpha : H_\Sigma(\mu\Sigma) \to \mu\Sigma$. In other words, $\mu\Sigma$ is the initial Σ-algebra (see (1)).

Since $\mu\Sigma$ is the colimit of the λ-chain $\mathcal{D}$ of Set^S defined in Theorem 14.7, Theorem 6.4 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 Σ-algebra. The unique Σ-homomorphism $fold^{\mathcal{A}} : \mu\Sigma \to A$ is the unique S-sorted function such that

$$\coprod_{i<\lambda} \mathcal{D}(i) \stackrel{[\beta_i]_{i<\lambda}}{\to} A \;=\; \coprod_{i<\lambda} \mathcal{D}(i) \stackrel{nat_\sim}{\to} \mu\Sigma \stackrel{fold^{\mathcal{A}}}{\to} A$$

where β_0 is the unique S-sorted function from $\mathcal{D}(0)$ to A and for all $i < \lambda$ and $s \in S$,

$$\beta_{i+1,s} = [f^{\mathcal{A}} \circ F_e(\beta_{i,s})]_{f:e\to s\in F} : \mathcal{D}(i+1)_s \to A_s.$$

H_Σ is ω-cocontinuous and its object mapping reads as follows:

For all S-sorted sets A and $s \in S$,

$$\begin{aligned} H_\Sigma(A)_s &= \textstyle\coprod_{f:e_1\times\cdots\times e_n\to s\in F} \prod_{i=1}^n A_{e_i} \\ &= \{((a_1,\dots,a_n),f) \mid f : e_1\times\cdots\times e_n \to s \in F,\ a_i \in A_{e_i},\ 1 \le i \le n\}. \end{aligned}$$

Hence for all $s \in S$, $k \in \mathbb{N}$ and $t \in \mathcal{D}(k)$,

$$\begin{aligned}\mathcal{D}(0)_s &= \emptyset,\\ \mathcal{D}(k+1)_s &= H_\Sigma(\mathcal{D}(k))_s\\ &= \{((t_1,\dots,t_n),f) \mid f: e_1\times\dots\times e_n \to s\in F,\ t_i\in\mathcal{D}(k)_{e_i},\ 1\le i\le n\},\\ \mathcal{D}(k,k+1)(t) &= t,\end{aligned}$$

and thus by Theorem 6.4,

$$\mu\Sigma_s \;=\; (\coprod_{k\in\mathbb{N}}\mathcal{D}(k)_s)/{\sim_s} \;\cong\; \bigcup_{k\in\mathbb{N}}\mathcal{D}(k)_s$$

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 Lemma 14.1 (1), $\alpha: H_\Sigma(A)\to A$ as defined in diagram (1) is iso and thus for all $f: e_1\times\dots\times e_n\to s\in F$ and $t_i\in\mu\Sigma_{e_i}$, $1\le i\le n$,

$$f^{\mu\Sigma}(t_1,\dots,t_n) = ((t_1,\dots,t_n),f).$$

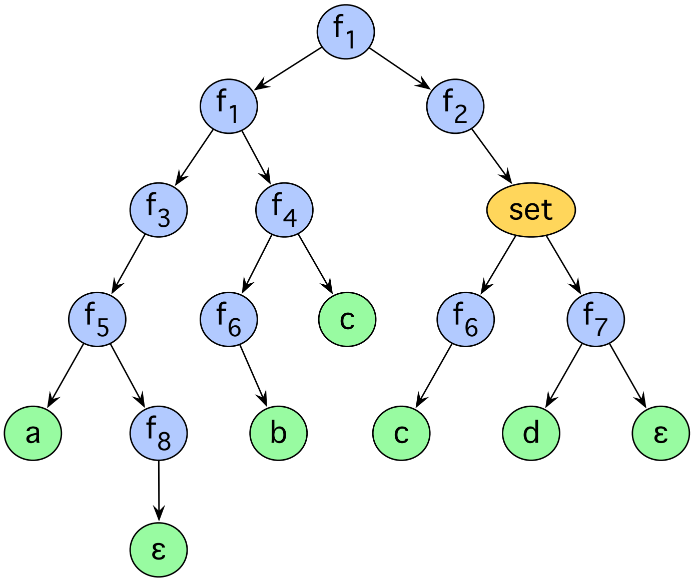
Hence for all Σ-algebras A,

$$fold^{\mathcal{A}}(((t_1,\dots,t_n),f)) = fold^{\mathcal{A}}(f^{\mu\Sigma}(t_1,\dots,t_n)) = f^{\mathcal{A}}(fold^{\mathcal{A}}_{e_1}(t_1),\dots,fold^{\mathcal{A}}_{e_n}(t_n)).$$

Moreover, for $A = \mu\Sigma$ and all $s \in S$,

$$A_s \cong H_\Sigma(A)_s = \{((a_1, \ldots, a_n), f) \mid f : e_1 \times \cdots \times e_n \to s \in F,\ a_i \in A_{e_i},\ 1 \leq i \leq n\}.$$

Hence $\mu\Sigma$ can be represented as a quotient of T_Σ (see section 19.12).



A ground Σ-term with constructors $f_1, \ldots, f_8$ and base elements a, b, c, d, ϵ.

Final models of destructive polynomial signatures

Let $\Sigma = (S, F)$ be a destructive polynomial signature, κ be the cardinality of the greatest exponent occurring in the range of some $f \in F$ and λ be the first regular cardinal number $> \kappa$.

By Theorem 14.6 (1) implies that H_Σ is λ-continuous and thus by Theorem 14.8, $coAlg_{H_\Sigma}$ has a final object $\alpha : \nu\Sigma \to H_\Sigma(\nu\Sigma)$. In other words, $\nu\Sigma$ is the final Σ-algebra (see (1)).

Since $\nu\Sigma$ is the limit of the ω-cochain $\mathcal{D}$ of Set^S defined in Theorem 14.8, Theorem 6.2 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 Σ-algebra. The unique Σ-homomorphism $unfold^{\mathcal{A}} : A \to \nu\Sigma$ is the unique S-sorted function such that

$$A \stackrel{\langle\beta_i\rangle_{i<\omega}}{\longrightarrow} \prod_{i<\omega} \mathcal{D}(i) \quad = \quad A \stackrel{unfold^{\mathcal{A}}}{\longrightarrow} \nu\Sigma \stackrel{inc}{\longrightarrow} \prod_{i<\omega} \mathcal{D}(i)$$

where β_0 is the unique S-sorted function from A to $\mathcal{D}(0)$ and for all $i < \omega$ and $s \in S$,

$$\beta_{i+1,s} = \langle F_e(\beta_{i,s}) \circ f^{\mathcal{A}} \rangle_{f:s\to e\in F} : A_s \to \mathcal{D}(i+1)_s.$$

H_Σ is ω-continuous and its object mapping reads as follows:

For all S-sorted sets A and $s \in S$,

$$\begin{aligned} H_\Sigma(A)_s &= \textstyle\prod_{f:s\to(e_1+\dots+e_n)^X\in F}(\coprod_{i=1}^n A_{e_i})^X \\ &= \{t : F \to \textstyle\bigcup_{f:s\to(e_1+\dots+e_n)^X\in F}(\coprod_{i=1}^n A_{e_i})^X \mid \\ &\qquad\qquad \forall\, f : s \to (e_1+\dots+e_n)^X \in F : t(f) \in (\textstyle\coprod_{i=1}^n A_{e_i})^X\} \\ &= \{t : F \to (A \times \mathbb{N})^X \mid \forall\, f : s \to (e_1+\dots+e_n)^X \in F\ \forall\, x \in X \\ &\qquad\qquad \exists\, 1 \le i \le n : t(f)(x) \in A_{e_i} \times \{i\}\}. \end{aligned}$$

Hence for all $s \in S$, $k \in \mathbb{N}$, $t \in \mathcal{D}(k+1)$ and $f \in F$,

$$\begin{aligned}
\mathcal{D}(0)_s &= 1, \\
\mathcal{D}(k+1)_s &= H_\Sigma(\mathcal{D}(k))_s = \{t : F \to (\mathcal{D}(k) \times \mathbb{N})^X \mid \\
&\qquad \forall\, f : s \to (e_1 + \cdots + e_n)^X \in F \;\forall\, x \in X \\
&\qquad \exists\, 1 \leq i \leq n : t(f)(x) \in \mathcal{D}(k)_{e_i} \times \{i\}\}, \\
\mathcal{D}(k+1,k)(t)(f) &= \pi_1 \circ t(f),
\end{aligned}$$

and thus by Theorem 6.2,

$$\begin{aligned}
\nu\Sigma_s &= \{t \in \textstyle\prod_{k\in\mathbb{N}} \mathcal{D}(k)_s \mid \forall\, k \in \mathbb{N}\; \forall\, f \in F : \mathcal{D}(k+1,k)(\pi_{k+1}(t))(f) = \pi_k(t)(f)\} \\
&= \{t \in \textstyle\prod_{k\in\mathbb{N}} \mathcal{D}(k)_s \mid \forall\, k \in \mathbb{N}\; \forall\, f \in F : \pi_1 \circ \pi_{k+1}(t)(f) = \pi_k(t)(f)\}.
\end{aligned}$$

18.1 Bounded functors

Let $\alpha : A \to F(A)$ be an F-coalgebra and B be a subset of A. If the inclusion mapping $inc : B \to A$ is a $coAlg_F$-morphism from an F-coalgebra $\beta : B \to F(B)$ to α then β is an F**-invariant** or F**-subcoalgebra** of α.

Theorem 18.1 ([81], 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 $\langle B\rangle : C \to F(C)$ such that C includes B. ❑

Let M be an S-sorted set. $F : Set^S \to Set^S$ is M**-bounded** if for all F-coalgebras $\alpha : A \to F(A)$ and $a \in A$, $|\langle a\rangle_s| \leq |M_s|$ (see [59], section 4).

This section aims at Theorem 18.4, which tells us that for all M-bounded functors F, Alg_F has a final object.

Let λ be a cardinal number.

A category $\mathcal{I}$ is **λ-filtered** if for each class $\mathcal{L}$ of less than λ $\mathcal{I}$-objects there is a cocone $\{i \to j \mid i \in \mathcal{L}\}$ in $\mathcal{I}$ and for all $\mathcal{I}$-objects i, j and each set Φ of less than λ $\mathcal{I}$-morphisms from i to j there is a coequalizing $\mathcal{I}$-morphism $h : j \to k$, i.e., for all $f, g \in \Phi$, $h \circ f = h \circ g$.

A diagram $\mathcal{D} : \mathcal{I} \to \mathcal{K}$ is **λ-filtered** if $\mathcal{I}$ is a λ-filtered category.

A functor $F : \mathcal{K} \to \mathcal{L}$ is **λ-accessible** if F preserves the colimits of all λ-filtered diagrams $\mathcal{D} : \mathcal{I} \to \mathcal{K}$ (see [10], section 5.2).

Theorem 18.2 ([11], Thm. 4.1; [12], 5.3)

Let M be an S-sorted set. $F : Set^S \to Set^S$ is M-bounded if F is $|M|$-accessible. Conversely, F is $(|M| + 1)$-accessible if F is M-bounded. ❑

By [156], Thm. 10.6, or [59], Cor. 4.9, for every destructive signature Σ there is an S-sorted set M such that H_Σ is M-bounded (see chapter 15).

Examples

By [156], Ex. 6.8.2, or [59], 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 section 9.16). Hence $|\langle st \rangle| \leq |X^*|$.

$B_{NAut(X,Y)}$ (see chapter 15) is $(X^* \times \mathbb{N})$-bounded: For all $B_{NAut(X,Y)}$-algebras

$$A \stackrel{\langle \delta,\beta \rangle}{\to} B_{NAut(X,Y)}(A)$$

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 $id_A^\#(a)(x \cdot w) = \cup\{id_A^\#(st')(w) \mid st' \in \delta^{\mathcal{A}}(a)(x)\}$ for all $x \in X$ and $w \in X^*$. Since for all $a \in A_{state}$ and $x \in X$, $|\delta^{\mathcal{A}}(a)(x)| \in \mathbb{N}$, $|\langle st \rangle| \leq |X^* \times \mathbb{N}|$. If $X = 1$, then $X^* \times \mathbb{N} \cong \mathbb{N}$ and thus $B_{NAut(1,Y)}$ is $\mathbb{N}$-bounded (see [156], Ex. 6.8.1; [59], section 5.1). ❑

A destructive signature $\Sigma = (S, F)$ is **Moore-like** if there is an S-sorted set M such that for all $f : s \to e \in F$, $e = s^{M_s}$ or $e \in \mathcal{I}$. Then M is called the **input** of Σ.

Lemma 18.3

Let $\Sigma = (S, F)$ be a Moore-like signature with input M and

$$F' = \{f : s \to e \mid e \in BT\}.$$

Let κ be the cardinality of the greatest exponent occurring in the source of some $f \in F$ and λ be the first regular cardinal number $> \kappa$.

Since Σ is polynomial, Theorem 14.6 (1) implies that H_Σ is λ-continuous and thus by Theorem 14.8, $coAlg_{H_\Sigma}$ has a final object $\alpha : A \to H_\Sigma(A)$. In other words, Alg_Σ has a final object A.

Let $Y = \prod_{f:s \to e \in F'} e$. If $|S| = 1$, then Σ agrees with $DAut(M_s, Y)$ and thus

$$A \;\cong\; Beh(M_s, Y)$$

(see sample algebra 9.6.24).

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. ❏

??MOORETAU

Let $\Sigma = (S, F)$ be a destructive signature, M be an S-sorted set, H_Σ be M-bounded and

$$F' = \{f_s : s \rightarrow s^{M_s} \mid s \in S\} \cup \{f' : s \rightarrow M_e \mid f : s \rightarrow e \in F\}.$$

Of course, $\Sigma' = (S, F')$ is Moore-like.

Let the function $\tau : H_{\Sigma'} \to H_\Sigma$ be defined as follows: For all S-sorted sets A, $a \in H_{\Sigma'}(A)_s$ and $f : s \to e \in F$,

$$\pi_f(\tau_{A,s}(a)) = F_e(\pi_{f_s}(a))(\pi_{f'}(a)).$$

τ is a surjective natural transformation.

Proof. The theorem generalizes [59], Thm. 4.7 (i)⇒(iv), from Set to Set^S. ❑

Theorem 18.4

Let $\Sigma = (S, F)$ be a destructive signature, M be an S-sorted set, H_Σ be M-bounded and the Σ-algebra A be defined as follows: For all $s \in S$,

$$A_s \;=\; M_s^* \to \prod_{f:s\to e\in F} M_e,$$

and for all $f : s \to e \in F$ 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_Σ where $\sim$ is the greatest Σ-congruence on A.

Proof. Let Σ' and τ be defined as in Theorem ??. Let $Y = \prod_{f':s\rightarrow M_e\in F'} M_e$. Since Σ' is Moore-like, Lemma 18.3 implies that the following Σ'-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 15.3, 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 $h \in A_s$,

$$\begin{aligned} f^{\mathcal{A}}(h) &= F_e(\lambda x.\lambda w.h(x \cdot w))(\pi_f(h(\epsilon))) = F_e(\lambda x.\lambda w.h(x \cdot w))(\pi_{f'}(h(\epsilon))) \\ &= F_e(f_s^{\mathcal{A}}(h))(f'^{\mathcal{A}}(h)) = F_e(\pi_{f_s}(g_1(h), \ldots, g_n(h)))(\pi_{f'}(g_1(h), \ldots, g_n(h))) \\ &= \pi_f(\tau_{A,s}(g_1(h), \ldots, g_n(h))) = \pi_f(\tau_{A,s}(\langle g_1, \ldots, g_n\rangle(h))) = f^B(h) \end{aligned}$$

where $\{g_1, \ldots, g_n\} = \{g^{\mathcal{A}} \mid g : s \rightarrow e' \in F'\}$.

Hence again by Lemma 15.3, $A/{\sim}$ is final in Alg_Σ where $\sim$ is the greatest Σ-congruence on A.

A direct proof of the existence of a final Σ-algebra is given by [60], Thm. 3.5. ❏

Example

Let $\Sigma = NAut(X,Y)$, i.e., $S = \{state\}$,

$$F = \{\delta : state \to set(state)^X,\ \beta : state \to Y\}$$

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_Σ is M-bounded, Theorem 18.4 implies that the following Σ-algebra A is weakly final:

$$A_{state} \;=\; M^* \to \mathcal{P}_\omega(M)^X \times Y.$$

For all $h \in A_{state}$ and $x \in X$, $h(\epsilon) = (g, y)$ implies

$$\begin{aligned}
\delta^{\mathcal{A}}(h)(x) &= F_{set(state)^X}(\lambda m.\lambda w.h(mw))(\pi_\delta(h(\epsilon)))(x) \\
&= F_{set(state)^X}(\lambda m.\lambda w.h(mw))(g)(x) = F_{set(state)}(\lambda m.\lambda w.h(mw))(g(x)) \\
&= \{F_{state}(\lambda m.\lambda w.h(mw))(m) \mid m \in g(x)\} \\
&= \{\lambda m.\lambda w.h(mw))(m) \mid m \in g(x)\} = \{\lambda w.h(mw) \mid m \in g(x)\}, \\
\beta^{\mathcal{A}}(h) &= F_Y(\lambda x.\lambda w.h(x \cdot w))(\pi_\beta(h(\epsilon))) = F_Y(\lambda x.\lambda w.h(x \cdot w))(y) = id_Y(y) = y.
\end{aligned}$$

Moreover, $A/{\sim}$ is final in Alg_Σ where $\sim$ is the greatest Σ-congruence on A, i.e., the union of all S-sorted binary relations $\sim$ on A such that for all $h, h' \in A_{state}$,

$$h \sim h' \text{ implies } \delta^{\mathcal{A}}(h) \sim_{set(state)^X} \delta^{\mathcal{A}}(h') \wedge \beta^{\mathcal{A}}(h) \sim_Y \beta^{\mathcal{A}}(h'),$$

i.e., for all $x \in X$, $h \sim h'$, $h(\epsilon) = (g, y)$ and $h'(\epsilon) = (g', y')$ imply

$$\begin{array}{l} \forall\, m \in g(x)\ \exists n \in g'(x) : \lambda w.h(mw) \sim \lambda w.h'(nw) \ \wedge \\ \forall\, n \in g'(x)\ \exists m \in g(x) : \lambda w.h(mw) \sim \lambda w.h'(nw) \ \wedge\ y = y'. \end{array}$$

Let $F' = \{f : state \to state^M,\ \delta : state \to \mathcal{P}_\omega(M)^X,\ \beta : state \to Y\}$ and $\Sigma' = (S, \{X, Y, M, \mathcal{P}_\omega(M)^X\}, F')$.

A is constructed from the following Σ'-algebra B with $B_{state} = A_{state}$ (see the proof of Theorem 18.4): For all $h \in A_{state}$, $f^B_{state}(h) = \lambda m.\lambda w.h(mw)$ and $\langle \delta^B, \beta^B \rangle(h) = h(\epsilon)$.

Since Σ' is Moore-like, Lemma 18.3 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. [62], section 5, shows (for the case $X = Y = 1$) how these trees yield the quotient $A/{\sim}$. ❑

19 Adjunctions

19.1 Five equivalent definitions

Given two categories $\mathcal{K}$ and $\mathcal{L}$, an **adjunction from $\mathcal{K}$ to $\mathcal{L}$** is a quadruple (L, R, ϕ, ψ) consisting of functors $L : \mathcal{K} \to \mathcal{L}$, $R : \mathcal{L} \to \mathcal{K}$ and a $(\mathcal{K} \times \mathcal{L})$-sorted bijection

$$\phi = (\phi_{A,B} : \mathcal{K}(A, RB) \to \mathcal{L}(LA, B))_{A\in\mathcal{K}, B\in\mathcal{L}}$$

with inverse ψ such that for all $A \in \mathcal{K}$ and $B \in \mathcal{L}$, $\phi_{A,B}$ is **natural in A and B**, i.e., for all $f : A \to RB$, $a : A' \to A \in \mathcal{K}$ and $g : LA \to B$, $b : B \to B' \in \mathcal{L}$,

$$\phi_{A',B}(f \circ a) = \phi_{A,B}(f) \circ La, \quad (1)$$

$$\phi_{A,B'}(Rb \circ f) = b \circ \phi_{A,B}(f), \quad (2)$$

or, equivalently,

$$\psi_{A,B'}(b \circ g) = Rb \circ \psi_{A,B}(g), \quad (3)$$

$$\psi_{A',B}(g \circ La) = \psi_{A,B}(g) \circ a \quad (4)$$

We write $L \dashv R$ and call L a **left adjoint** of R and R a **right adjoint** of L (see, e.g., [101], section IV.1; [16], Def. 7.3.7).

Left adjoints preserve colimits and thus initial objects.
Right adjoints preserve limits and thus final objects.

Theorem 19.1 ([16], Theorem 7.3.12)

The following five conditions are equivalent:

(i) There is an adjunction (L, R, ϕ, ψ) from $\mathcal{K}$ to $\mathcal{L}$ such that for all $f : A \to RB \in \mathcal{K}$ and $g : LA \to B \in \mathcal{L}$,

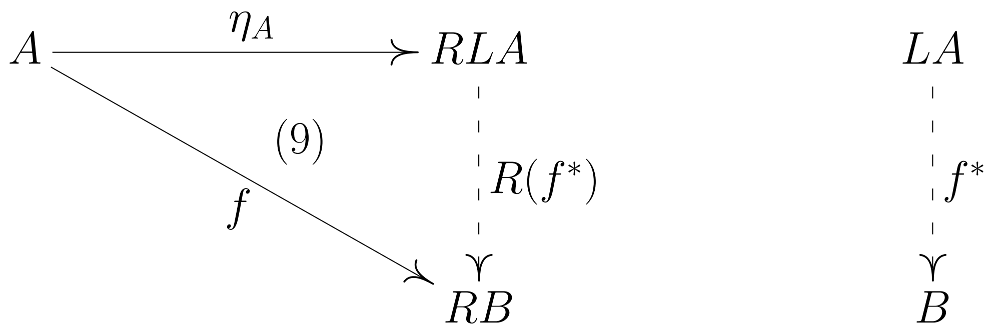
$$\phi_{A,B}(f) = \phi_{RB,B}(id_{RB}) \circ Lf, \quad (5)$$
$$\psi_{A,B}(g) = Rg \circ \psi_{A,LA}(id_{LA}). \quad (6)$$

(ii) There is an adjunction (L, R, ϕ, ψ) from $\mathcal{K}$ to $\mathcal{L}$.

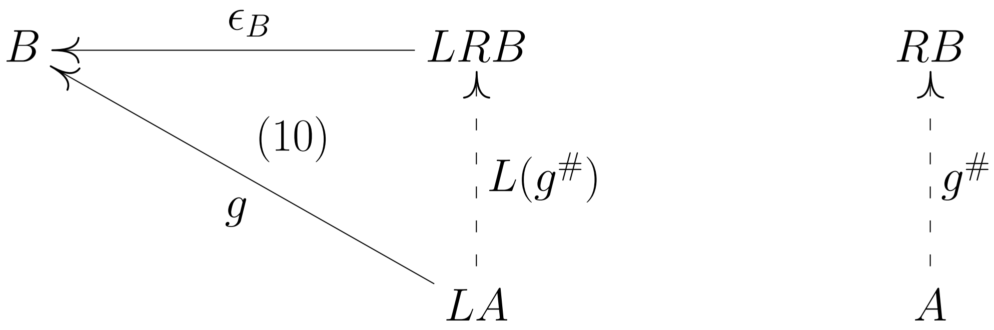
(iii) Given functors $L : \mathcal{K} \to \mathcal{L}$ and $R : \mathcal{L} \to \mathcal{K}$, there are natural transformations $\eta : Id_{\mathcal{K}} \to RL$, called **unit**, and $\epsilon : LR \to Id_{\mathcal{L}}$, called **co-unit**, such that for all $a \in \mathcal{K}$ and $B \in \mathcal{L}$,

$$\epsilon_{LA} \circ L\eta_A = id_{LA}, \quad (7)$$
$$R\epsilon_B \circ \eta_{RB} = id_{RB}. \quad (8)$$

(iv) Given a functor $R : \mathcal{L} \to \mathcal{K}$, for all $A \in \mathcal{K}$, there are an $\mathcal{L}$-object LA, called **free over** A, and a $\mathcal{K}$-morphism $\eta_A : A \to RLA$ such that for every $f : A \to RB \in \mathcal{K}$ there is a unique $\mathcal{L}$-morphism $f^* : LA \to B$, called the **left adjunct** or **$\mathcal{L}$-extension of** f, such that the following diagram commutes:



(v) Given a functor $L : \mathcal{K} \to \mathcal{L}$, for all $B \in \mathcal{L}$, there are a $\mathcal{K}$-object RB, called **cofree over** B, and an $\mathcal{L}$-morphism $\epsilon_B : LRB \to B$ such that for every $g : LA \to B \in \mathcal{L}$ there is a unique $\mathcal{K}$-morphism $g^\# : A \to RB$, called the **right adjunct** or **$\mathcal{K}$-coextension of** g, such that the following diagram commutes:



Proof. "(ii)⇒(iii)": Let $\psi = \phi^{-1}$. For all $A \in \mathcal{K}$ and $B \in \mathcal{L}$, define

$$\epsilon_B = \phi_{RB,B}(id_{RB}) : LRB \to B, \tag{11}$$

$$\eta_A = \psi_{A,LA}(id_{LA}) : A \to RLA. \tag{12}$$

ϵ is natural: For all $h : B' \to B \in \mathcal{L}$,

$$\epsilon_B \circ LRh \overset{(11)}{=} \phi_{RB,B}(id_{RB}) \circ LRh \overset{(1)}{=} \phi_{RB',B}(id_{RB} \circ Rh) = \phi_{RB',B}(Rh)$$
$$= \phi_{RB',B}(Rh \circ id_{RB'}) \overset{(2)}{=} h \circ \phi_{RB',B'}(id_{RB'}) \overset{(11)}{=} h \circ \epsilon_{B'}.$$

η is natural: For all $h : A \to A' \in \mathcal{L}$,

$$RLh \circ \eta_A \overset{(12)}{=} RLh \circ \psi_{A,LA}(id_{LA}) \overset{(3)}{=} \psi_{A,LA'}(Lh \circ id_{LA}) = \psi_{A,LA'}(Lh)$$
$$= \psi_{A,LA'}(id_{LA'} \circ Lh) \overset{(4)}{=} \psi_{A',LA'}(id_{LA'}) \circ h \overset{(12)}{=} \eta_{A'} \circ h.$$

(7) holds true:

$$\epsilon_{LA} \circ L\eta_A = \phi_{RLA,LA}(id_{RLA}) \circ L(\psi_{A,LA}(id_{LA})) \overset{(1)}{=} \phi_{A,LA}(id_{RLA} \circ \psi_{A,LA}(id_{LA})) = id_{LA}.$$

(8) holds true:

$$R\epsilon_B \circ \eta_{RB} = R(\phi_{RB,B}(id_{RB})) \circ \psi_{RB,LRB}(id_{LRB}) \overset{(3)}{=} \psi_{RB,B}(\phi_{RB,B}(id_{RB}) \circ id_{LRB}) = id_{RB}.$$

"(iii)⇒(iv)+(v)": For all $f : A \to RB$ and $g : LA \to B$, define

$$f^* = \epsilon_B \circ Lf : LA \to B, \tag{13}$$

$$g^\# = Rg \circ \eta_A : A \to RB. \tag{14}$$

Hence

$$R(f^*) \circ \eta_A \overset{(13)}{=} R(\epsilon_B \circ Lf) \circ \eta_A = R\epsilon_B \circ RLf \circ \eta_A = R\epsilon_B \circ \eta_{RB} \circ f \overset{(6)}{=} f, \tag{15}$$

$$\epsilon_B \circ L(g^\#) \overset{(14)}{=} \epsilon_B \circ L(Rg \circ \eta_A) = \epsilon_B \circ LRg \circ L\eta_A = g \circ \epsilon_{LA} \circ L\eta_A \overset{(5)}{=} g. \tag{16}$$

Moreover, $\_^* : \mathcal{K}(A, RB) \to \mathcal{L}(LA, B)$ and $\_^\# : \mathcal{L}(LA, B) \to \mathcal{K}(A, RB)$ are inverse to each other:

$$\begin{aligned}(f^*)^\# &\overset{(13)}{=} (\epsilon_B \circ Lf)^\# \overset{(14)}{=} R(\epsilon_B \circ Lf) \circ \eta_A = R\epsilon_B \circ RLf \circ \eta_A \\ &= R\epsilon_B \circ \eta_{RB} \circ f \overset{(8)}{=} f,\end{aligned} \tag{17}$$

$$(g^{\#})^* \overset{(14)}{=} (Rg \circ \eta_A)^* \overset{(13)}{=} \epsilon_B \circ L(Rg \circ \eta_A) = \epsilon_B \circ LRg \circ L\eta_A \\ = g \circ \epsilon_{LA} \circ L\eta_A \overset{(7)}{=} g. \tag{18}$$

Hence $\mathcal{L}$-extensions and $\mathcal{K}$-coextensions are unique:

Let $g : LA \to B \in \mathcal{K}$ satisfy $Rg \circ \eta_A = f$. Then $g \overset{(18)}{=} (g^{\#})^* = (Rg \circ \eta_A)^* = f^*$.

Let $f : A \to RB \in \mathcal{K}$ satisfy $\epsilon_B \circ Lf = g$. Then $f \overset{(17)}{=} (f^*)^{\#} = (\epsilon_B \circ Lf)^{\#} = g^{\#}$.

"(iv)⇒(i)": L is functor from $\mathcal{K}$ to $\mathcal{L}$: For all $a : A' \to A \in \mathcal{K}$, define

$$La = (\eta_A \circ a)^* : LA' \to LA. \tag{19}$$

Consequently, $\eta = (\eta_A : A \to RLA)_{A\in\mathcal{K}}$ is a natural transformation:

$$RLa \circ \eta_{A'} = R((\eta_A \circ a)^*) \circ \eta_{A'} = \eta_A \circ a. \tag{20}$$

For all $A \in \mathcal{K}$ and $B \in \mathcal{L}$, define $\phi_{A,B} : \mathcal{K}(A, RB) \to \mathcal{L}(LA, B)$ and $\psi_{A,B} : \mathcal{L}(LA, B) \to \mathcal{K}(A, RB)$ as follows: For all $f : A \to RB \in \mathcal{K}$ and $g : LA \to B \in \mathcal{L}$,

$$\phi_{A,B}(f) = f^* : LA \to B, \tag{21}$$

$$\psi_{A,B}(g) = Rg \circ \eta_A : A \to RB. \tag{22}$$

ϕ and ψ are inverse to each other:

$$\psi_{A,B}(\phi_{A,B}(f)) \overset{(21)}{=} \psi_{A,B}(f^*) \overset{(22)}{=} R(f^*) \circ \eta_A \overset{(9)}{=} f,$$

$$R(\phi_{A,B}(\psi_{A,B}(g))) \circ \eta_A \overset{(22)}{=} R(\phi_{A,B}(Rg \circ \eta_A)) \circ \eta_A \overset{(21)}{=} R((Rg \circ \eta_A)^*) \circ \eta_A \overset{(9)}{=} Rg \circ \eta_A$$

and thus $\phi_{A,B}(\psi_{A,B}(g)) = g$ by the uniqueness part of (9).

(1) holds true:

$$R(\phi_{A',B}(f \circ a)) \circ \eta_{A'} \overset{(22)}{=} R((f \circ a)^*) \circ \eta_{A'} \overset{(9)}{=} f \circ a \overset{(9)}{=} R(f^*) \circ \eta_A \circ a$$
$$\overset{(20)}{=} R(f^*) \circ RLa \circ \eta_{A'} = R(f^* \circ La) \circ \eta_{A'}$$

and thus $\phi_{A',B}(f \circ a) = f^* \circ La \overset{(21)}{=} \phi_{A,B}(f) \circ La$ by the uniqueness part of (9).

(2) holds true:

$$R(\phi_{A,B'}(Rb \circ f)) \circ \eta_A \overset{(22)}{=} R((Rb \circ f)^*) \circ \eta_A \overset{(9)}{=} Rb \circ f \overset{(9)}{=} Rb \circ R(f^*) \circ \eta_A$$
$$= R(b \circ f^*) \circ \eta_A \overset{(21)}{=} R(b \circ \phi_{A,B}(f)) \circ \eta_A$$

and thus $\phi_{A,B'}(Rb \circ f) = b \circ \phi_{A,B}(f)$ by the uniqueness part of (9).

(5) holds true:

$$\phi_{A,B}(f) \overset{(21)}{=} f^* = (id_{RB} \circ f)^* \overset{(23)}{=} id_{RB}^* \circ (\eta_{RB} \circ f)^* \overset{(21),(19)}{=} \phi_{RB,B}(id_{RB}) \circ Lf$$

where (23) follows from the uniqueness part of (9).

(6) holds true:

$$\psi_{A,B}(g) \overset{(22)}{=} Rg \circ \eta_A = Rg \circ id_{RLA} \circ \eta_A = Rg \circ R(id_{LA}) \circ \eta_A \overset{(22)}{=} Rg \circ \psi_{A,LA}(id_{LA}).$$

"(v)⇒(i)": R is functor from $\mathcal{L}$ to $\mathcal{K}$: For all $a : B \to B' \in \mathcal{L}$, define

$$Rb = (b \circ \epsilon_B)^{\#} : LA' \to LA. \tag{24}$$

Consequently, $\epsilon = (\epsilon_B : B \to LRB)_{B\in\mathcal{L}}$ is a natural transformation:

$$\epsilon_{B'} \circ LRb = \epsilon_{B'} \circ L((b \circ \epsilon_B)^{\#}) = b \circ \epsilon_B. \tag{25}$$

For all $A \in \mathcal{K}$ and $B \in \mathcal{L}$, define $\phi_{A,B} : \mathcal{K}(A, RB) \to \mathcal{L}(LA, B)$ and $\psi_{A,B} : \mathcal{L}(LA, B) \to \mathcal{K}(A, RB)$ as follows: For all $f : A \to RB \in \mathcal{K}$ and $g : LA \to B \in \mathcal{L}$,

$$\phi_{A,B}(f) = \epsilon_B \circ Lf : LA \to B, \tag{26}$$

$$\psi_{A,B}(g) = g^{\#} : A \to RB. \tag{27}$$

ϕ and ψ are inverse to each other:

$$\phi_{A,B}(\psi_{A,B}(g)) \overset{(27)}{=} \phi_{A,B}(g^{\#}) \overset{(26)}{=} \epsilon_B \circ L(g^{\#}) \overset{(10)}{=} g,$$

$$\epsilon_B \circ L(\psi_{A,B}(\phi_{A,B}(f))) = \epsilon_B \circ L(\psi_{A,B}(\epsilon_B \circ Lf)) \overset{(27)}{=} \epsilon_B \circ L((\epsilon_B \circ Lf)^{\#}) \overset{(10)}{=} \epsilon_B \circ Lf,$$

and thus $\psi_{A,B}(\phi_{A,B}(f)) = f$ by the uniqueness part of (10).

(3) holds true:

$$\epsilon_{B'} \circ L(\psi_{A,B'}(b \circ g)) \overset{(27)}{=} \epsilon_{B'} \circ L((b \circ g)^{\#}) \overset{(10)}{=} b \circ g \overset{(10)}{=} b \circ \epsilon_B \circ L(g^{\#})$$
$$\overset{(25)}{=} \epsilon_{B'} \circ LRb \circ L(g^{\#}) = \epsilon_{B'} \circ L(Rb \circ g^{\#})$$

and thus $\psi_{A,B'}(b \circ g) = Rb \circ g^{\#} \overset{(27)}{=} Rb \circ \psi_{A,B}(g)$ by the uniqueness part of (10).

(4) holds true:

$$\epsilon_{B'} \circ L(\psi_{A,B'}(g \circ La)) \overset{(27)}{=} \epsilon_{B'} \circ L((g \circ La)^{\#}) \overset{(10)}{=} g \circ La \overset{(10)}{=} \epsilon_B \circ L(g^{\#}) \circ La$$
$$= \epsilon_B \circ L(g^{\#} \circ a) \overset{(27)}{=} \epsilon_{B'} \circ L(\psi_{A,B}(g) \circ a)$$

and thus $\psi_{A,B'}(g \circ La) = \psi_{A,B}(g) \circ a$ by the uniqueness part of (10).

(5) holds true:

$$\phi_{A,B}(f) \overset{(26)}{=} \epsilon_B \circ Lf = \epsilon_B \circ id_{LRB} \circ Lf = \epsilon_B \circ L(id_{RB}) \circ Lf \overset{(26)}{=} \phi_{RB,B}(id_{RB}) \circ Lf.$$

(6) holds true:

$$\psi_{A,B}(g) \overset{(27)}{=} g^{\#} = (g \circ id_{LA})^{\#} \overset{(28)}{=} (g \circ \epsilon_{LA})^{\#} \circ id_{LA}^{\#} \overset{(24),(27)}{=} Rg \circ \psi_{A,LA}(id_{LA}).$$

where (28) follows from the uniqueness part of (10).

Finally, we show the equivalence of (1)+(2) and (3)+(4):

"(2)⇒(3)":

$$\psi_{A,B'}(b \circ g) = \psi_{A,B'}(b \circ \phi_{A,B}(\psi_{A,B}(g))) \overset{(2)}{=} \psi_{A,B'}(\phi_{A,B'}(Rb \circ \psi_{A,B}(g))) = Rb \circ \psi_{A,B}(g).$$

"(1)⇒(4)":

$$\psi_{A',B}(g \circ La) = \psi_{A',B}(\phi_{A,B}(\psi_{A,B}(g)) \circ La) \overset{(1)}{=} \psi_{A',B}(\phi_{A',B}(\psi_{A,B}(g) \circ a)) = \psi_{A,B}(g) \circ a.$$

"(4)⇒(1)":

$$\phi_{A',B}(f \circ a) = \phi_{A',B}(\psi_{A,B}(\phi_{A,B}(f)) \circ a) \overset{(4)}{=} \phi_{A',B}(\psi_{A',B}(\phi_{A,B}(f) \circ La)) = \phi_{A,B}(f) \circ La.$$

"(3)⇒(2)":

$$\phi_{A,B'}(Rb \circ f) = \phi_{A,B'}(Rb \circ \psi_{A,B}(\phi_{A,B}(f))) \overset{(2)}{=} b \circ \phi_{A,B}(\psi_{A,B}(\phi_{A,B}(f))) = b \circ \phi_{A,B}(f).$$ ❏

No matter which one of the above five ways define a particular adjunction, (11)-(14), (19), (21), (22), (24), (26) and (27) provide us with valid relationships, which allow us to obtain L, R, ϕ, ψ, the unit, the co-unit, $\mathcal{L}$-extensions and $\mathcal{K}$-coextensions from each other. Moreover, (17) and (18) motivate a rule-like notation:

$$\frac{A \xrightarrow{f} RB}{LA \xrightarrow{f^*} B} \qquad\qquad \frac{LA \xrightarrow{g} B}{A \xrightarrow{g^\#} RB}$$

19.2 Identity functor

The identity functor $Id_{\mathcal{K}} : \mathcal{K} \to \mathcal{K}$ is left and right adjoint to itself.

$$\frac{A \xrightarrow{f} Id_{\mathcal{K}}(B)}{A \xrightarrow{f^* = f} B} \qquad \frac{Id_{\mathcal{K}}(A) \xrightarrow{g} B}{A \xrightarrow{g^{\#} = g} B}$$

19.3 Monoid functor

Let *Monoid* be the full subcategory of Alg_{Mon} whose objects are monoids.

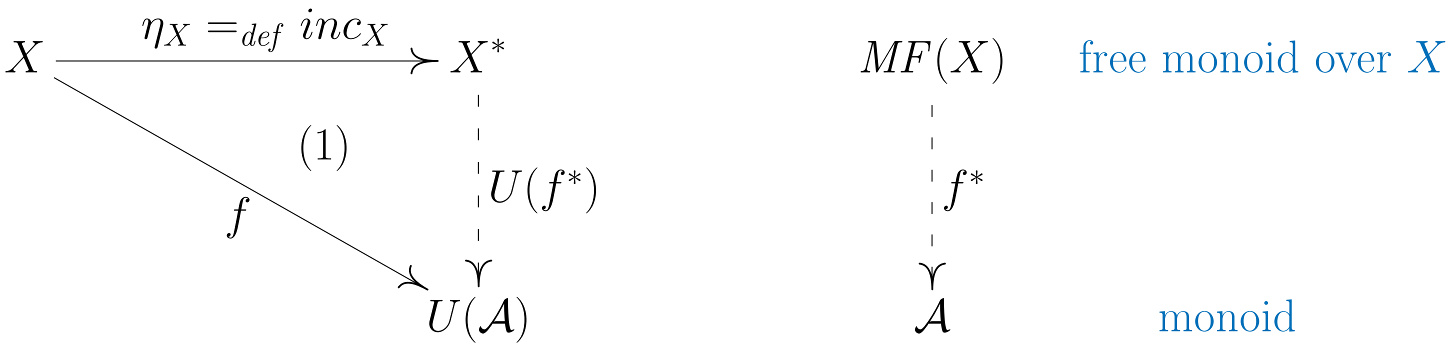
The **monoid functor**

$$\begin{array}{rcl} MF : Set & \to & Monoid \subseteq Alg_{Mon} \\ X & \mapsto & (X^*, \{\lambda\epsilon.\epsilon : 1 \to X^*, \lambda(v,w).vw : (X^*)^2 \to X^*\}) \\ f : X \to A & \mapsto & MF(f) : X^* \to A^* \end{array}$$

(see chapter 8; [16], section 7.2) where for all $x \in X$ and $w \in X^*$,

$$MF(f)(\epsilon) =_{def} \epsilon,$$
$$MF(f)(x \cdot w) =_{def} f(x) \cdot MF(f)(w),$$

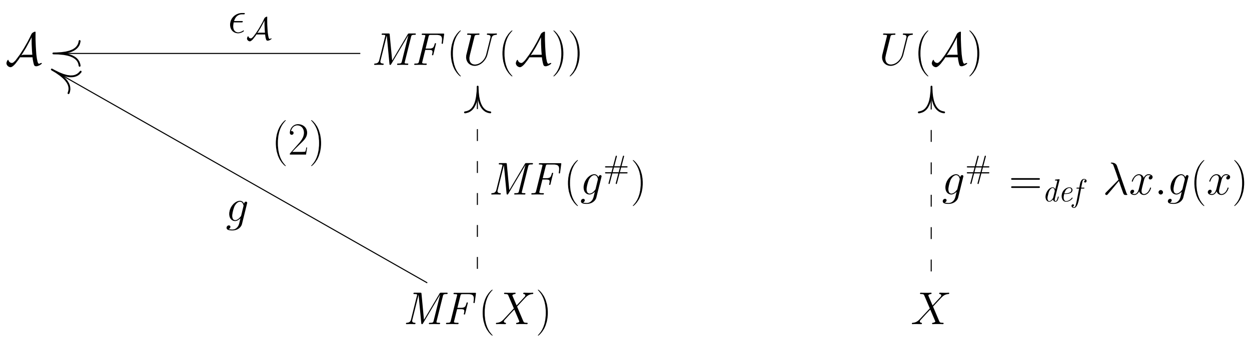
is left adjoint to the forgetful functor $U : Monoid \to Set$ (which maps a monoid to its carrier), i.e., for all monoids $\mathcal{A}$ and functions $f : X \to U(\mathcal{A})$ there is a unique Mon-homomorphism $f^* : MF(X) \to \mathcal{A}$ such that (1) commutes:



For all $x \in X$ and $w \in X^*$,

$$f^*(\epsilon) =_{def} one^{\mathcal{A}},$$
$$f^*(x \cdot w) =_{def} mul^{\mathcal{A}}(f(x), f^*(w)).$$

Equivalently, for all monoids $\mathcal{A}$ and Mon-homomorphisms $g : MF(X) \to \mathcal{A}$ there is a unique function $g^\# : X \to U(\mathcal{A})$ such that (2) commutes:



For all $a \in U(\mathcal{A})$ and $w \in U(\mathcal{A})^*$,

$$\begin{aligned} \epsilon_{\mathcal{A}}(\epsilon) &=_{def} one^{\mathcal{A}}, \\ \epsilon_{\mathcal{A}}(aw) &=_{def} mul^{\mathcal{A}}(a, \epsilon_{\mathcal{A}}(w)). \end{aligned}$$

Summing up:

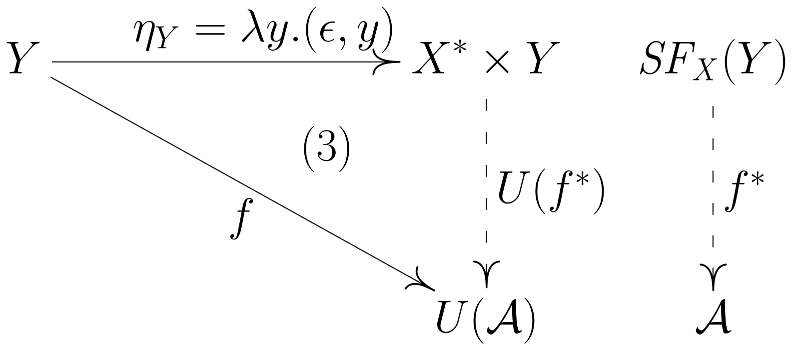
$$\frac{X \xrightarrow{f} U(\mathcal{A})}{MF(X) \xrightarrow{f^*} \mathcal{A}} \qquad\qquad \frac{MF(X) \xrightarrow{g} \mathcal{A}}{X \xrightarrow{g^\#} U(\mathcal{A})}$$

19.4 Sequence functor

The **sequence functor**

$$\begin{aligned} SF_X : Set &\rightarrow Alg_{coStream(X)} \\ Y &\mapsto (X^* \times Y, \{cons^{Seq(X,Y)}\}) \\ f : Y \rightarrow A &\mapsto \lambda(w,y).(w, f(y)) : X^* \times Y \rightarrow X^* \times A \end{aligned}$$

(see sample algebra 9.6.3; [16], section 7.2) is left adjoint to the forgetful functor $U : Alg_{coStream(X)} \rightarrow Set$, 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:



free $coStream(X)$-algebra over Y

$=$ initial $Dyn(X,Y)$-algebra with $\alpha^{SF_X(Y)} = \eta_Y$

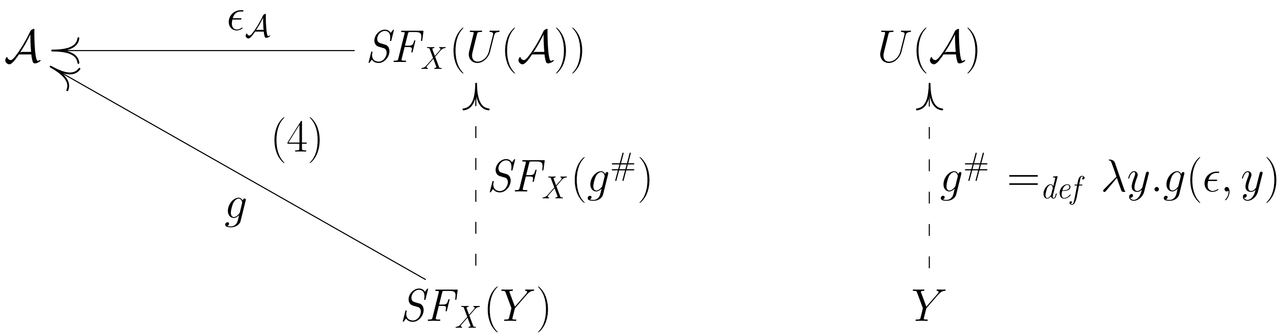
$coStream(X)$-algebra

$=$ $Dyn(X,Y)$-algebra with $\alpha^{\mathcal{A}} = f$

For all $y \in Y$, $x \in X$ and $w \in X^*$,

$$f^*(\epsilon, y) =_{def} f(y),$$
$$f^*(xw, y) =_{def} cons^{\mathcal{A}}(x, f^*(w, y)).$$

Equivalently, for all $coStream(X)$-algebras $\mathcal{A}$ and $coStream(X)$-homomorphisms $g : SF_X(Y) \to \mathcal{A}$ there is a unique function $g^\# : Y \to U(\mathcal{A})$ such that (4) commutes:



For all $a \in U(\mathcal{A})$, $x \in X$ and $w \in X^*$,

$$\epsilon_{\mathcal{A}}(\epsilon, a) =_{def} a,$$
$$\epsilon_{\mathcal{A}}(xw, a) =_{def} cons^{\mathcal{A}}(x, \epsilon_{\mathcal{A}}(w, a)).$$

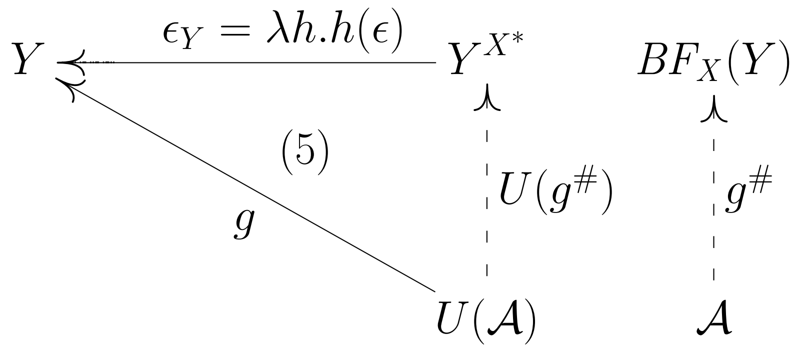
Summing up:

$$\frac{Y \xrightarrow{f} U(\mathcal{A})}{SF_X(Y) \xrightarrow{f^*} \mathcal{A}} \qquad\qquad \frac{SF_X(Y) \xrightarrow{g} \mathcal{A}}{Y \xrightarrow{g^\#} U(\mathcal{A})}$$

19.5 Behavior functor

$$\begin{aligned} BF_X : Set &\rightarrow Alg_{Med(X)} \\ Y &\mapsto (Y^{X^*}, \{\delta^{Beh(X,Y)}\}) \\ g : A \rightarrow Y &\mapsto \lambda h.g \circ h : A^{X^*} \rightarrow Y^{X^*} \end{aligned}$$

(see sample algebra 9.6.24; [16], section 7.2; [17], section 3.1) 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:



cofree $Med(X)$-algebra over Y

= final $DAut(X,Y)$-algebra

with $\beta^{BF_X(Y)} = \epsilon_Y$

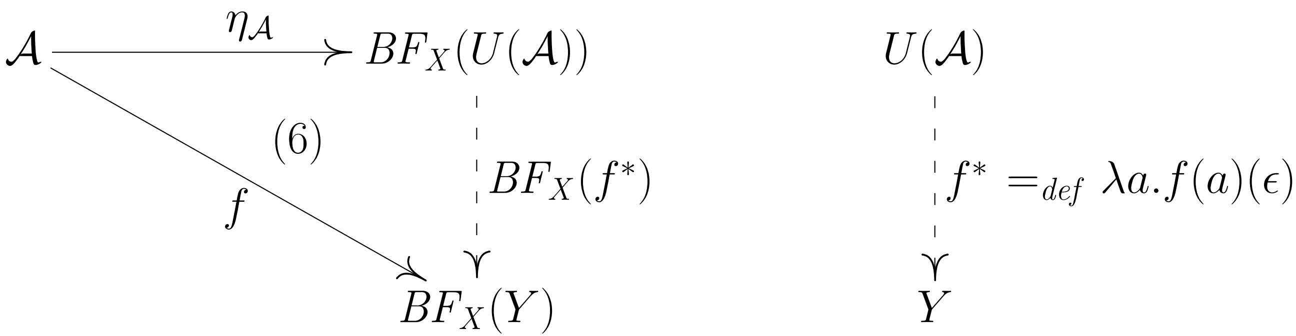
$Med(X)$-algebra

= $DAut(X,Y)$-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 $f : \mathcal{A} \to BF_X(Y)$ there is a unique function $f^* : U(\mathcal{A}) \to Y$ such that (6) commutes:



For all $a \in U(\mathcal{A})$, $x \in X$ and $w \in X^*$,

$$\eta_{\mathcal{A}}(a)(\epsilon) =_{def} a,$$
$$\eta_{\mathcal{A}}(a)(x \cdot w) =_{def} \eta_{\mathcal{A}}(\delta^{\mathcal{A}}(a)(x))(w).$$

Summing up:

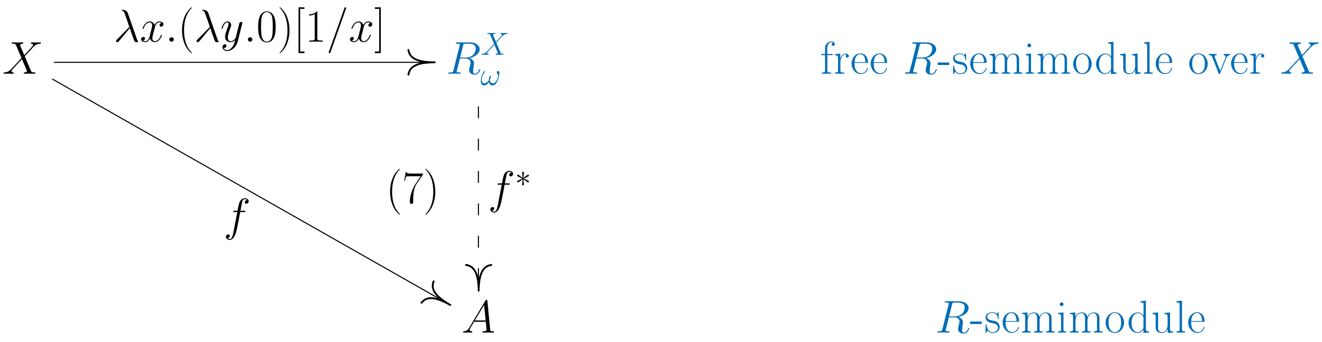
$$\frac{\mathcal{A} \xrightarrow{f} BF_X(Y)}{U(\mathcal{A}) \xrightarrow{f^*} Y} \qquad \frac{U(\mathcal{A}) \xrightarrow{g} Y}{\mathcal{A} \xrightarrow{g^\#} BF_X(Y)}$$

19.6 Weighted-set functor

Let $(R, +, 0, *, 1)$ be a semiring (see sample algebra 9.6.25). The weighted-set functor $R_\omega^- : Set \to SMod_R$ is left adjoint to the forgetful functor $U : SMod_R \to Set$.

For all R-semimodules A with R-action $\cdot : R \times A \to A$ and functions $f : X \to A$ there is a unique linear function $f^* : R_\omega^X \to A$ such that (7) commutes:

For all $g \in R_R^X$, $f^*(g) =_{def} \sum_{x \in supp(g)} g(x) \cdot f(x)$.



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:

$$
\begin{aligned}
Lin(\mathcal{A})_{state} &= R_\omega^A, \\
\delta^{Lin(\mathcal{A})} &= (\delta^{\mathcal{A}})^* : R_\omega^A \to (R_\omega^A)^X, \\
\beta^{Lin(\mathcal{A})} &= (\beta^{\mathcal{A}})^* : R_\omega^A \to R.
\end{aligned}
$$

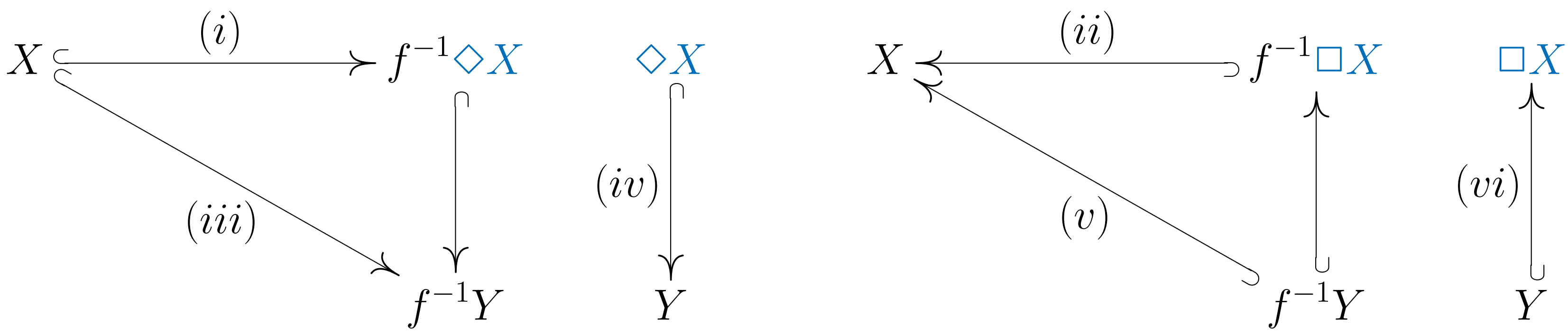
19.7 Box and diamond functors

Let A, B be sets, $f : A \to B$ 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

$$\begin{aligned} f^{-1} : \mathcal{P}(B) &\to \mathcal{P}(A) \\ Y &\mapsto \{a \in A \mid f(a) \in Y\}. \end{aligned}$$

The following functors are left or right adjoint to f^{-1}:

$$\begin{aligned} \Diamond : \mathcal{P}(A) &\to \mathcal{P}(B) \\ X &\mapsto \{b \in B \mid f^{-1}(b) \cap X \neq \emptyset\} \\ &= f(X) = \{f(x) \mid x \in X\} \end{aligned} \qquad \begin{aligned} \Box : \mathcal{P}(A) &\to \mathcal{P}(B) \\ X &\mapsto \{b \in B \mid f^{-1}(b) \subseteq X\} \end{aligned}$$



Proof.

$$(i) \quad X \subseteq \{a \in A \mid f(a) \in f(X) = \Diamond X\} = f^{-1}\Diamond X$$

$$(ii) \quad f^{-1}\Box X = \{a \in A \mid f(a) \in \Box X = \{b \in B \mid f^{-1}(b) \subseteq X\}\}$$

$$= \{a \in A \mid f^{-1}(f(a)) \subseteq X\} = \{a \in A \mid \{a' \in A \mid f(a') = f(a)\} \subseteq X\}$$

$$= \{a \in A \mid \forall\, a' \in A : f(a') = f(a) \Rightarrow a' \in X\} \subseteq X$$

$$(iii) \quad X \subseteq f^{-1}Y$$

$$\Rightarrow \Diamond X = \{b \in B \mid f^{-1}(b) \cap X \neq \emptyset\} \subseteq \{b \in B \mid f^{-1}(b) \cap f^{-1}Y \neq \emptyset\}$$

$$\Leftrightarrow \Diamond X \subseteq \{b \in B \mid \{a \in A \mid f(a) = b\} \cap \{a \in A \mid f(a) \in Y\} \neq \emptyset\}$$

$$\Leftrightarrow \Diamond X \subseteq \{b \in B \mid \{a \in A \mid f(a) = b \in Y\} \neq \emptyset\}$$

$$\Leftrightarrow (iv) \quad \Diamond X \subseteq \{b \in B \mid \{a \in A \mid f(a) = b \in Y\} \neq \emptyset\} \subseteq Y$$

$$(v) \quad f^{-1}Y \subseteq X$$

$$\Rightarrow \{b \in B \mid f^{-1}(b) \subseteq f^{-1}Y\} \subseteq \{b \in B \mid f^{-1}(b) \subseteq X\} = \Box X$$

$$\Leftrightarrow \{b \in B \mid \{a \in A \mid f(a) = b\} \subseteq \{a \in A \mid f(a) \in Y\}\} \subseteq \Box X$$

$$\Leftrightarrow \{b \in B \mid \forall\, a \in A : (f(a) = b \Rightarrow f(a) \in Y)\} \subseteq \Box X$$

$$\Leftrightarrow (vi) \quad Y \subseteq \{b \in B \mid \forall\, a \in A : (f(a) = b \Rightarrow f(a) \in Y)\} \subseteq \Box X$$

19.8 Strongly connected components

Let $G \in Alg_{Graph}$ (see chapter 8). $(e_0, \ldots, e_{n-1}) \in G^+_{edge}$ is a **cycle of** G if for all $0 \leq i < n$, $target^G(e_i) = source^G(e_{(i+1) \; mod \; n})$. $cycles(G)$ denotes the set of cycles of G.

Let $S = \{node, edge\}$. The S-sorted equivalence relation $\sim$ relates all strongly connected nodes of G to each other: For all $e, e' \in G_{edge}$ and $a, b \in G_{node}$,

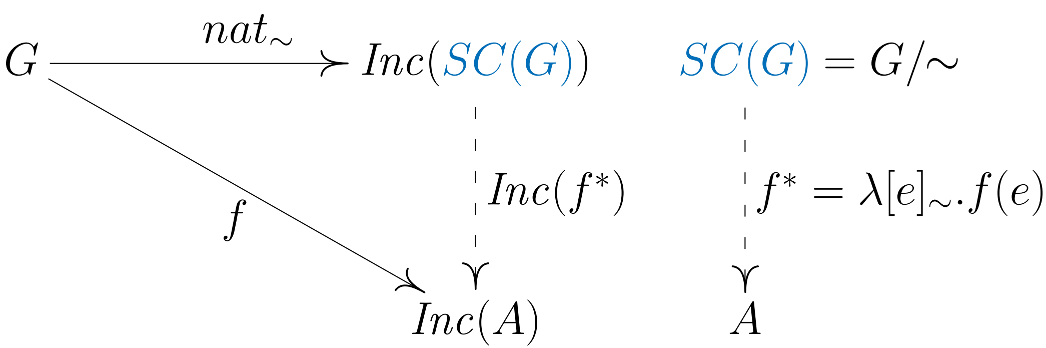
$$a \sim_{node} b \;\; \Leftrightarrow_{def} \;\; a = b \vee \begin{cases} \exists \; (e_0, \ldots, e_{n-1}) \in cycles(G), \;\; 0 \leq i, j < n : \\ a = source^G(e_i) \wedge b = source^G(e_j), \end{cases}$$

$$e \sim_{edge} e' \;\; \Leftrightarrow_{def} \;\; e = e' \vee \exists \; (e_0, \ldots, e_{n-1}) \in cycles(G), \;\; 0 \leq i, j < n : e = e_i \wedge e' = e_j.$$

$\sim_G = (\sim_{node}, \sim_{edge})$ is a $Graph$-congruence, i.e., for all $e, e' \in G_{edge}$, $e \sim_{edge} e'$ implies $source^G(e) \sim_{node} source^G(e')$ and $target^G(e) \sim_{node} target^G(e')$.

Let $Acyclic$ be the full subcategory of Alg_{Graph} whose objects are the acyclic graphs.

The functor $SC : Alg_{Graph} \to Acyclic$ 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 $Inc : Acyclic \to Alg_{Graph}$ ([144], Ex. 2.4.10).



strongly connected components of G

acyclic graph

f^* is well-defined: Let $e \sim_{edge} e'$. If $f_{edge}(e) \neq f_{edge}(e')$, then $e \neq e'$ and thus g has a cycle $(e_0, \ldots, e_{n-1})$. Since f is a *Graph*-homomorphic, for all $0 \leq i < n$,

$$target^{\mathcal{A}}(f_{edge}(e_i)) = f_{edge}(target^G(e_i)) = f_{edge}(source^G(e_{(i+1)\ mod\ n}))$$
$$= source^{\mathcal{A}}(f_{edge}(e_{(i+1)\ mod\ n})).$$

Hence $(f_{edge}(e_0), \ldots, f_{edge}(e_{n-1}))$ is a cycle of the acyclic graph A. $\lightning$
We conclude $f_{edge}(e) = f_{edge}(e')$.

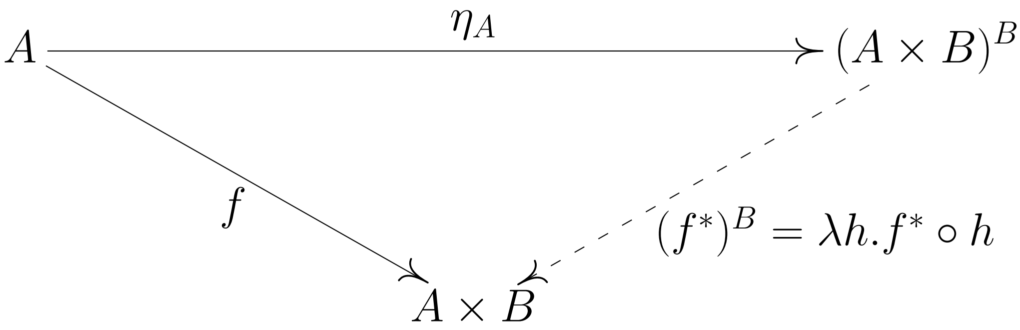
Let $a \sim_{node} b$. If $f_{node}(a) \neq f_{node}(b)$, then $a \neq b$ and thus G has a cycle. As above we obtain a contradiction and thus conclude $f_{node}(a) = f_{node}(b)$.

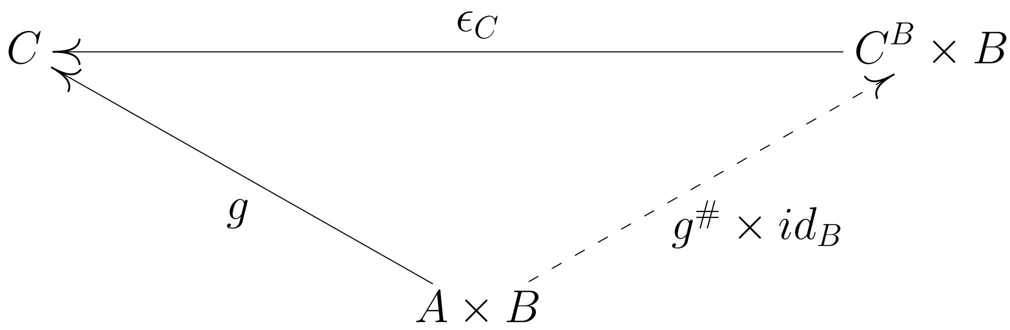
Since $nat_{\sim_G}$ and $f = f^* \circ nat_{\sim_G}$ are graph homomorphisms and $nat_{\sim_G}$ is surjective, Lemma 9.1 (1) implies that f^* is also a graph homomorphism. Hence the uniqueness of f^* satisfying $f = f^* \circ nat_{\sim_G}$ again follows from the fact that $nat_{\sim_G}$ is epi.

19.9 Reader and writer

Reader functors are left adjoint to writer functors.

$$\frac{A \xrightarrow{f} C^B}{A \times B \xrightarrow{f^*} C} \qquad \frac{A \times B \xrightarrow{g} C}{A \xrightarrow{g^\#} C^B}$$





This example suggests a notion of adjoint signatures Σ and Σ' with the same set of sorts:

Σ is **left** (**right**) **adjoint to** Σ' if H_Σ is left (right) adjoint to $H_{\Sigma'}$ (in $Mod(S)$; see chapter 15).

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 $\delta : state \times X \to state$ in $\mathcal{A}$ and $\delta : state \to state^X$ in $\mathcal{B}$ are obtained as the extension of $\delta^{\mathcal{A}}$ and the coextension of $\delta^{\mathcal{B}}$, respectively:

$$\frac{A \xrightarrow{\delta^{\mathcal{A}}} A^X}{A \times X \xrightarrow{(\delta^{\mathcal{A}})^*} A} \qquad\qquad \frac{A \times X \xrightarrow{\delta^{\mathcal{B}}} A}{A \xrightarrow{(\delta^{\mathcal{B}})^\#} A^X}$$

19.10 Cartesian closure and fixpoints

A category $\mathcal{K}$ is **Cartesian closed** if $\mathcal{K}$ has a final object Fin, binary products and for all $B \in \mathcal{K}$, the functor $\_ \times B$ that maps $A \in \mathcal{K}$ to $A \times B$ and $f : A \to B \in \mathcal{K}$ to $f \times id_B : A \times B \to B \times B$ has a right adjoint.

Unit, co-unit, $\mathcal{L}$-extensions and $\mathcal{K}$-coextensions of the corresponding adjunction are also denoted by $pair$, $apply$, $uncurry(f) : A \times B \to C$ (for all $f : A \to C^B \in \mathcal{K}$) and $curry(g) : A \to C^B$ (for all $g : A \times B \to C \in \mathcal{K}$), respectively.

These notations are inspired by the definitions that render Set and Set^S Cartesian closed categories:

Let A, B, C be S-sorted sets.

- $C^B =_{def} Set^S(B, C)$.
- For all S-sorted functions $f : A \to C$ and $g : B \to A$, $f^B(g) =_{def} f \circ g$.
- $\eta_A = pair_A =_{def} \lambda a.\lambda b.(a, b)$ and $\epsilon_C = apply_C = \lambda(f, b).f(b)$.
- For all S-sorted functions $g : A \times B \to C$, $g^{\#} = curry(g) =_{def} \lambda a.\lambda b.g(a, b)$.
- For all S-sorted functions $f : A \to C^B$, $f^* = uncurry(f) =_{def} \lambda(a, b).h(a)(b)$.

Let $\mathcal{K}$ be a Cartesian closed subcategory of Set^S.

$C \in \mathcal{K}$ **has the fixpoint property** if every $f : C \to C \in \mathcal{K}$ has a fixpoint (see section 3.2).

Theorem 19.2 (Cantor's Diagonal Theorem; [169], Thm. 1)

If there is a surjective $\mathcal{K}$-morphism $h : A \to C^A$, then C has the fixpoint property.

Proof. Let $f : C \to C \in \mathcal{K}$, $h : A \to C^A \in \mathcal{K}$ be surjective and $g = f \circ h^* \circ \Delta_A : A \to C$. Since h is surjective, there is $a_g \in A$ such that for all $a \in A$, $h(a_g)(a) = g(a)$. Hence

$$h(a_g)(a_g) = g(a_g) = f(h^*(a_g, a_g)) = f(h(a_g)(a_g)),$$

i.e., $h(a_g)(a_g)$ is a fixpoint of f. ❑

Theorem 19.2 can be generalized to arbitrary Cartesian closed categories:

Let $\mathcal{K}$ be a Cartesian closed category.

$C \in \mathcal{K}$ **has the fixpoint property** if for every $f : C \to C$ there is $c : Fin \to C$ with $f \circ c = c$.

A $\mathcal{K}$-morphism $h : A \to C^B$ is **weakly point-surjective** if for every $g : B \to C \in \mathcal{K}$ there is $a_g : Fin \to A$ such that for all $b : Fin \to B \in \mathcal{K}$, $\epsilon_C \circ \langle h \circ a_g, b \rangle = g \circ b$.

Theorem 19.3 (Lawvere's Diagonal Theorem; [99], Thm. 1.1)

If there is a weakly point-surjective $\mathcal{K}$-morphism $h : A \to C^A$, then C has the fixpoint property.

Proof. Let $f : C \to C \in \mathcal{K}$, $h : A \to C^A \in \mathcal{K}$ be weakly point-surjective and $g = f \circ h^* \circ \Delta_A : A \to C$. Since h is weakly point-surjective, there is $a_g : Fin \to A$ such that for all $a : Fin \to A \in \mathcal{K}$, $\epsilon_C \circ \langle h \circ a_g, a\rangle = g \circ a$. Hence

$$\epsilon_C \circ \langle h \circ a_g, a_g\rangle = g \circ a_g = f \circ h^* \circ \Delta_A \circ a_g = f \circ h^* \circ \langle a_g, a_g\rangle$$
$$\overset{Theorem\ 19.1\,(13)}{=} f \circ \epsilon_C \circ L(h) \circ \langle a_g, a_g\rangle = f \circ \epsilon_C \circ (h \times id_A) \circ \langle a_g, a_g\rangle$$
$$\overset{section\ 2.2\,(8)}{=} f \circ \epsilon_C \circ \langle h \circ a_g, id_A \circ a_g\rangle = f \circ \epsilon_C \circ \langle h \circ a_g, a_g\rangle,$$

i.e., $\epsilon_C \circ \langle h \circ a_g, a_g\rangle$ is a fixpoint of f. ❑

Corollary 19.4 (Diagonal Theorem for retractions)

If there is a retraction $h : A \to C^A \in \mathcal{K}$ (see section 4.2), then C has the fixpoint property.

Proof. Let $h : A \to C^A \in \mathcal{K}$ be a retraction, i.e., $h \circ h' = id_{A^C}$ for some $h' : A^C \to A \in \mathcal{K}$. By Theorem 19.3, it is sufficient to show that h is weakly point-surjective. Let $g : B \to C \in \mathcal{K}$ and $a_g = h' \circ (g \circ \pi_2)^\# : Fin \to A$. Then for all $b : Fin \to B \in \mathcal{K}$,

$$\epsilon_C \circ \langle h \circ a_g, b\rangle = \epsilon_C \circ \langle h \circ h' \circ (g \circ \pi_2)^\#, b\rangle = \epsilon_C \circ \langle (g \circ \pi_2)^\#, b\rangle$$
$$\overset{section\ 2.2\,(8)}{=} \epsilon_C \circ ((g \circ \pi_2)^\# \times id_B) \circ \langle id_{Fin}, b\rangle = g \circ \pi_2 \circ \langle id_{Fin}, b\rangle = g \circ b,$$

i.e., h is weakly point-surjective. ❑

Corollary 19.5

For all sets A, C with $|C| > 1$, $|A| < |C^A|$. In particular, $2^{\mathbb{N}}$ is uncountable.

Proof. Of course, $|A| \leq |C^A|$. Let $c, d \in C$ with $c \neq d$. Then $g : C \to C$ with $g(c) = d$ and $g(e) = c$ for all $e \in C \setminus \{c\}$ does not have a fixpoint. Hence by Theorem 19.4, there is no surjective $h : A \to C^A$ and thus $A \not\cong C^A$.

A fixpoint argument is also used in the following result:

Proposition 19.6 (Russell's paradox)

Let $\mathcal{K}$ be the category of classes, A be the class of all sets and $\chi : A \to 2^A$ be the function that maps each subclass B of A to its characteristic function $\chi(B) : A \to 2$. χ is not surjective. In particular, the class B of all sets S with $S \notin S$ is not a set.

Proof. Let $f = \lambda S.g(\chi(S)(S)) : A \to 2$ where $g : 2 \to 2$ maps 0 to 1 and 1 to 0. Then for all sets S,

$$f(S) = g(\chi(S)(S)) = 1 \;\Leftrightarrow\; \chi(S)(S) = 0 \;\Leftrightarrow\; S \notin S.$$

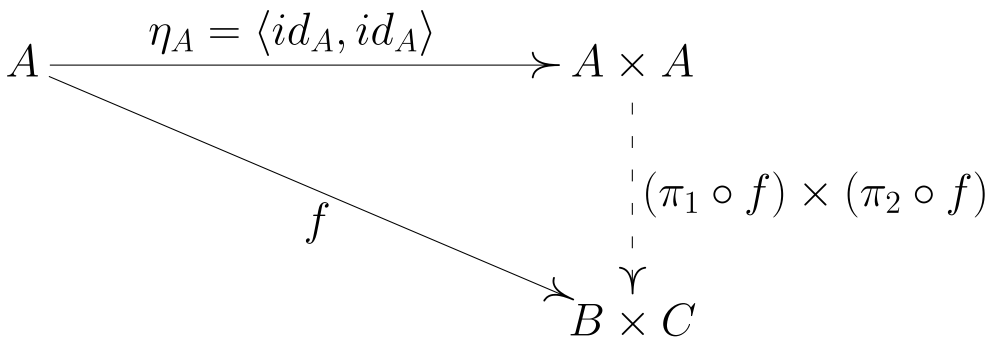
Consequently, if the collection T of all sets S with $S \notin S$ were a set, then $\chi(T) = f$ and thus $\chi(T)(T) = f(T) = g(\chi(T)(T))$, i.e., $\chi(T)(T)$ were a fixpoint of f, which is not possible. Hence T is not a set and χ is not surjective. ❏

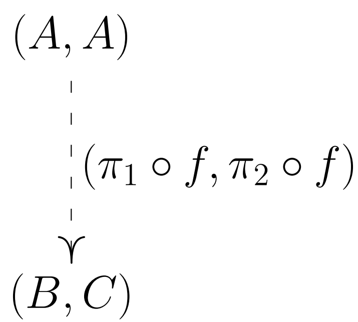
[169], section 1.3, and [192], §3 and §5, employ similar arguments for reformulating other "negative" results, like the unsolvability of the halting problem (Turing), the incompleteness of arithmetic theories (Gödel) or the undefinability of truth (Tarski).

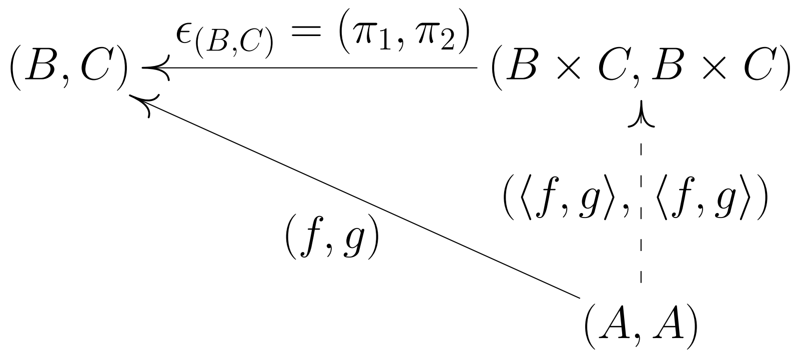
19.11 Product and coproduct

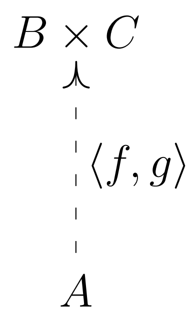
Products (and other limits) are right adjoint to diagonals.

$$\frac{A \xrightarrow{f} B \times C}{(A, A) \xrightarrow{f^* = (\pi_1 \circ f, \pi_2 \circ f)} (B, C)} \qquad \frac{(A, A) \xrightarrow{(f,g)} (B, C)}{A \xrightarrow{(f,g)^{\#} = \langle f, g \rangle} B \times C}$$

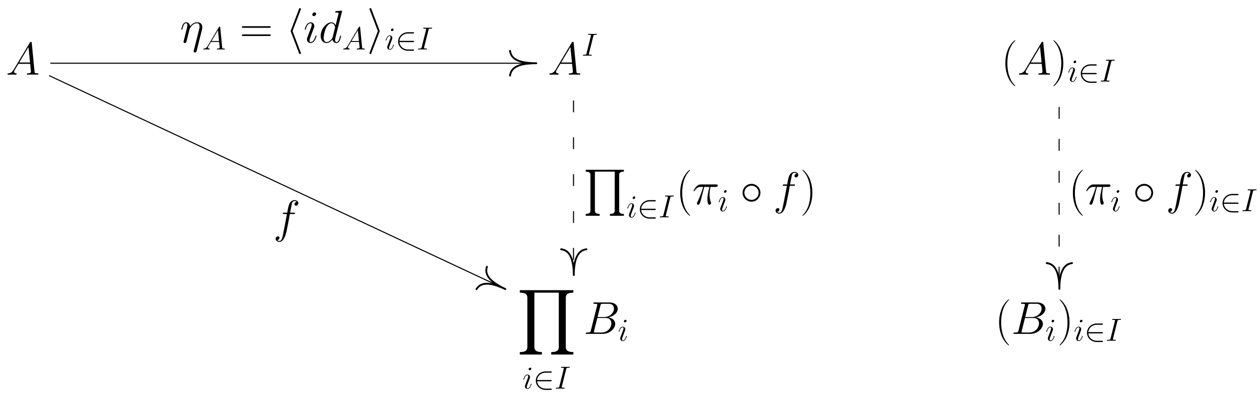


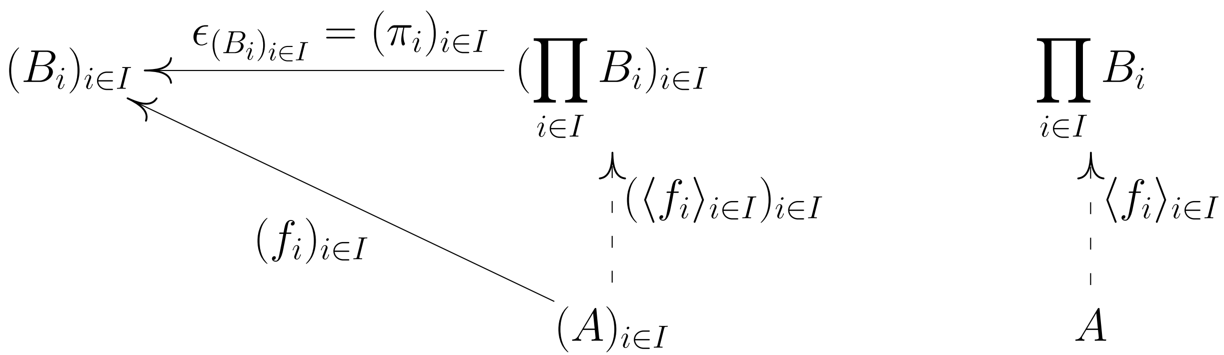






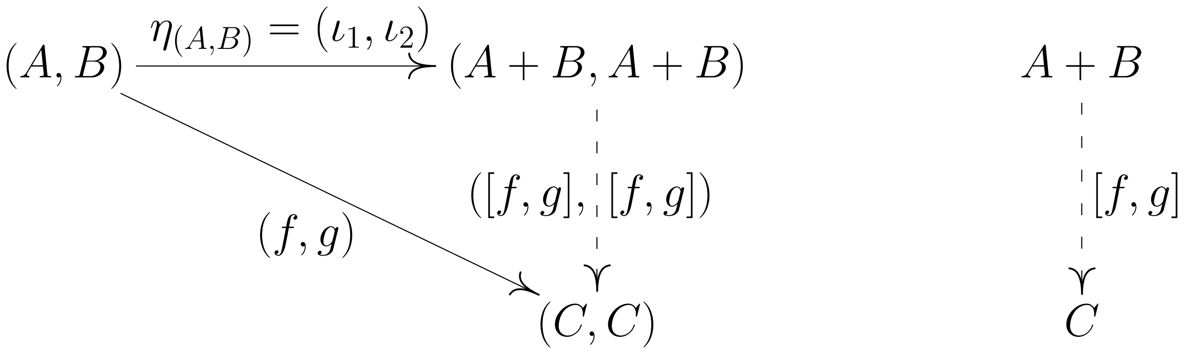
$R : \mathcal{K}^I \to \mathcal{K}$ with $R((B_i)_{i\in I}) = \prod_{i\in I} B_i$ for all $\mathcal{K}^I$-objects and -morphisms $(B_i)_{i\in I}$ is right adjoint to the diagonal functor $\Delta^I_{\mathcal{K}} : \mathcal{K} \to \mathcal{K}^I$ (see section 5).

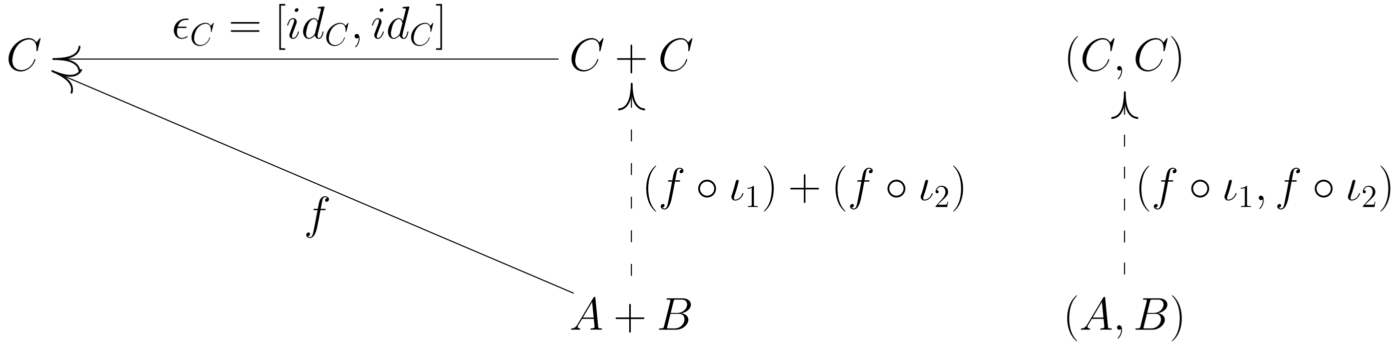




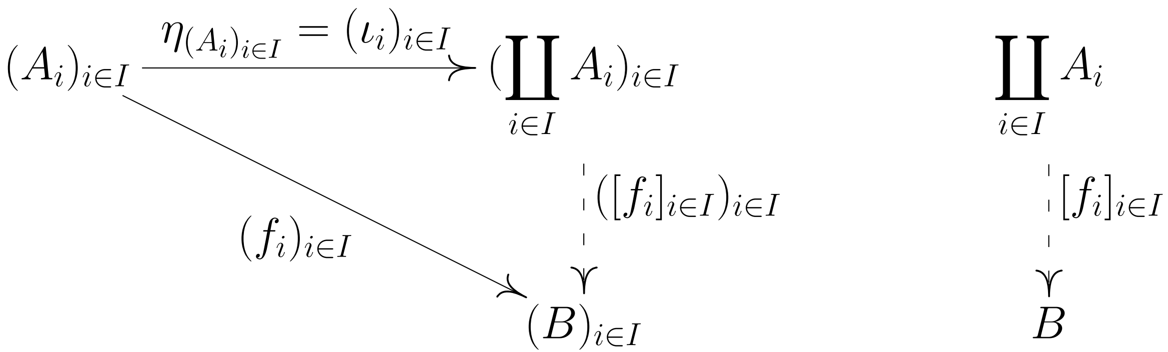
Coproducts (and other colimits) are left adjoint to diagonals.

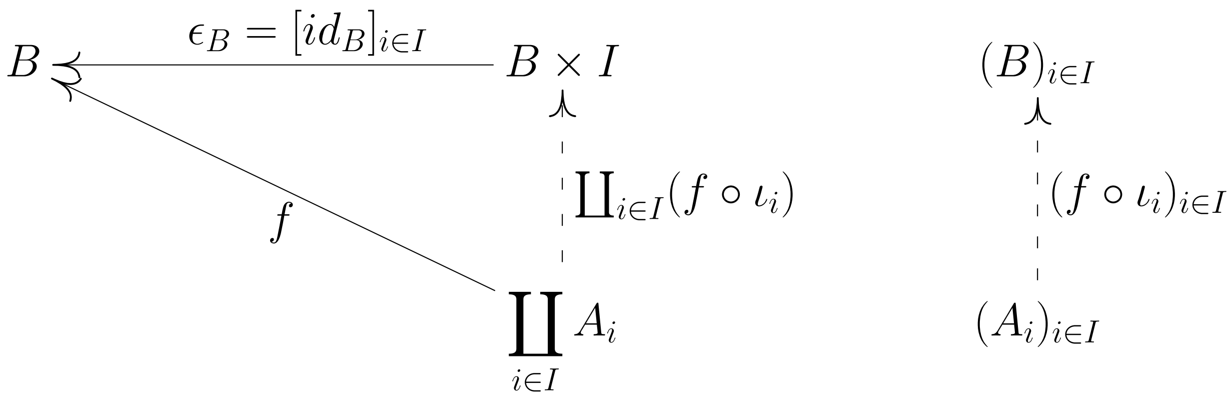
$$\frac{(A,B) \xrightarrow{(f,g)} (C,C)}{A+B \xrightarrow{(f,g)^* = [f,g]} C} \qquad \frac{A+B \xrightarrow{f} C}{(A,B) \xrightarrow{f^\# = (f\circ\iota_1, f\circ\iota_2)} (C,C)}$$





$L : \mathcal{K}^I \to \mathcal{K}$ 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 $\Delta^I_{\mathcal{K}} : \mathcal{K} \to \mathcal{K}^I$ (see section 5).





19.12 Term and flowchart functors

Let $\Sigma = (S, C)$ be a constructive polynomial signature, U_S be the forgetful functor from Alg_Σ to Set^S (see chapter 7), $V, V' \in Set^S$ and $\mathcal{A}$ be a Σ-algebra (see chapter 9).

$\eta_V =_{def} inc_V : V \to T_\Sigma(V)$.

$$\frac{V \xrightarrow{g} U_S(\mathcal{A})}{T_\Sigma(V) \xrightarrow{g^*} \mathcal{A}} \qquad\qquad \frac{T_\Sigma(V) \xrightarrow{h} \mathcal{A}}{V \xrightarrow{h^\# = h \circ \eta_V} U_S(\mathcal{A})}$$

The **term functor**

$$\begin{aligned} T_\Sigma : Set^S &\to Alg_\Sigma \\ V &\mapsto T_\Sigma(V) \\ f : V \to V' &\mapsto (\eta_{V'} \circ f)^* : T_\Sigma(V) \to T_\Sigma(V') \end{aligned}$$

is left adjoint to U_S. Proof of the functor property: Let $g : V' \to V''$. 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 9.7,

$$T_\Sigma(g \circ f) = (\eta_{V''} \circ g \circ f)^* = (\eta_{V''} \circ g)^* \circ (\eta_{V'} \circ f)^* = T_\Sigma(g) \circ T_\Sigma(f).$$

For all Σ-algebras $\mathcal{A}$ with carrier A, the co-unit $\epsilon_A = id_A^* : T_\Sigma(A) \to \mathcal{A}$ 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_Σ is initial in Alg_Σ, which also follows from the definition of the extension

$$fold^{\mathcal{A}} : T_\Sigma \to \mathcal{A}$$

(see section 9.11).

$T_\Sigma(V)$ represents $F_V =_{def} Set^S(V, U_S(\_))$ (see chapter 5) because for all $V \in Set^S$, the (covariant) functors F_V and $Alg_\Sigma(T_\Sigma(V), \_)$ are naturally equivalent, i.e., for all Σ-algebras $\mathcal{A}$, the set $F_V(\mathcal{A})$ of term valuations of V in (the carrier of) $\mathcal{A}$ and the set $Alg_\Sigma(T_\Sigma(V), \mathcal{A})$ of Σ-homomorphisms are isomorphic. Moreover, by Corollary 5.2 (7), this applies as well to isomorphic representations of $T_\Sigma(V)$.

For concrete representations of terms and valuations in the area of database schemas and schema instances, see [172], section 7.2.1.

Let $\Sigma(V)$ be defined as in section 9.12. 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, $fold^{\mathcal{A}} = (val^{\mathcal{A}})^*$. Hence by the uniqueness of $fold^{\mathcal{A}}$, $fold^{\mathcal{A}} = id_A^* \circ T_\Sigma(val^{\mathcal{A}})$.

For instance, let $\Sigma = coStream(X)$. Then $T_\Sigma \cong SF_X = X^* \times \_$ (see section 19.4) and $\Sigma(Y) = Dyn(X,Y)$. Hence $SF_X(Y)$ is initial in $Alg_{Dyn(X,Y)}$ and for all $Dyn(X,Y)$-algebras $\mathcal{A}$ with carrier A, $fold^{\mathcal{A}} = id_A^* \circ SF_X(\alpha^{\mathcal{A}}) = id_A^* \circ (id_{X^*} \times \alpha^{\mathcal{A}})$.

In particular, since $\Sigma(1) = Dyn(X,1) = List(X)$, $SF_X(1) \cong X^*$ is initial in $Alg_{List(X)}$ (see sample initial algebra 9.13 (3)).

From flowchart to state functors

Let $\Sigma = (S,D)$ be a destructive signature. The **flowchart functor**

$$\begin{aligned} \overline{T_\Sigma} : Set^S &\to Set^{\mathcal{T}_{po}(S)} \\ V &\mapsto \overline{T_\Sigma}(V) \\ f : V \to V' &\mapsto (inc_{V'} \circ f)^* : \overline{T_\Sigma}(V) \to \overline{T_\Sigma}(V') \end{aligned}$$

satisfies the functor property: Let $g : V' \to V''$. Then

$$(inc_{V''} \circ g)^* \circ (inc_{V'} \circ f)^* \circ inc_V = (inc_{V''} \circ g)^* \circ inc_{V'} \circ f = inc_{V''} \circ g \circ f.$$

Hence by Theorem 9.18,

$$T_\Sigma(g \circ f) = (\eta_{V''} \circ g \circ f)^* = (\eta_{V''} \circ g)^* \circ (\eta_{V'} \circ f)^* = T_\Sigma(g) \circ T_\Sigma(f).$$

Let $\mathcal{A}$ be a Σ-algebra with carrier A and $V \in Set^S$. The $\mathcal{T}_{po}(S)$-sorted function

$$\eta^\circ_V : T_{\overline{\Sigma}}(V) \to A_V$$

(see section 9.19) defines a natural transformation from the term functor $T_{\overline{\Sigma}}$ to the **state functor**

$$\begin{aligned} (\_ \times A)^A : Set^{\mathcal{T}_{po}(S)} &\to Set^{\mathcal{T}_{po}(S)} \\ V &\mapsto A_V \\ f : V \to V' &\mapsto \lambda g.(f \times A) \circ g : A_V \to A_{V'} \end{aligned}$$

$$\begin{array}{ccc} T_{\overline{\Sigma}}(V) & \xrightarrow{\eta^\circ_V} & (V \times A)^A \\ {\scriptstyle T_{\overline{\Sigma}}(f)} \downarrow & (1) & \downarrow {\scriptstyle (f \times A)^A} \\ T_{\overline{\Sigma}}(V') & \xrightarrow[\eta^\circ_{V'}]{} & (V' \times A)^A \end{array}$$

Proof of (1) by structural induction. Let $f \in Set^S(V, V')$.

$$(f \times A)^A(\eta^\circ_V()) = (f \times A)^A(id_1) = id_1(id_1) = id_1 = \eta^\circ_{V'}() = \eta^\circ_{V'}(id_1()) = \eta^\circ_{V'}(\overline{T_\Sigma}(f)()).$$

For all $s \in S$, $x \in V_s$ and $a \in A_s$,

$$(f \times A)^A(\eta^\circ_V(x))(a) = (f \times A)^A(\eta_V(x))(a) = (\lambda(z,a).(f(z),a) \circ \eta(x))(a)$$
$$= (\lambda(z,a).(f(z),a))(\eta(x)(a) = (\lambda(z,a).(f(z),a))(x,a) = (f(x),a) = \eta^\circ_{V'}(f(x))(a)$$
$$= \eta^\circ_{V'}((inc_{V'} \circ f)(x))(a) = \eta^\circ_{V'}((inc_{V'} \circ f)^*(x))(a) = \eta^\circ_{V'}(\overline{T_\Sigma}(f)(x))(a).$$

For all $d : s \to e \in D$, $t \in \overline{T_\Sigma}(V)_e$ and $a \in A_s$,

$$(f \times A)^A(\eta^\circ_V(d(t)))(a) = (f \times A)^A(\eta^\circ_V(t) \circ d^{\mathcal{A}})(a)$$
$$= (\lambda(x,a).(f(x),a) \circ \eta^\circ_V(t) \circ d^{\mathcal{A}})(a) = (\lambda(x,a).(f(x),a) \circ \eta^\circ_V(t))(d^{\mathcal{A}}(a))$$
$$\stackrel{ind.\ hyp.}{=} \eta^\circ_{V'}(\overline{T_\Sigma}(f)(t))(d^{\mathcal{A}}(a)) = \eta^\circ_{V'}(\overline{T_\Sigma}(f)(t) \circ d^{\mathcal{A}})(a) = \eta^\circ_{V'}(\overline{T_\Sigma}(f)(d(t)))(a).$$

For all $e = \coprod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $t = (t_i)_{i \in I} \in \times_{i \in I} \overline{T_\Sigma}(V)_{e_i}$, $i \in I$ and $a \in A_{e_i}$,

$$(f \times A)^A(\eta^\circ_V(t))(\iota_i(a)) = (\lambda(x,a).(f(x),a) \circ \eta^\circ_V(t))(\iota_i(a))$$
$$= (\lambda(x,a).(f(x),a) \circ [\eta^\circ_V(t_i)]_{i \in I})(\iota_i(a)) = [\lambda(x,a).(f(x),a) \circ \eta^\circ_V(t_i)]_{i \in I}(\iota_i(a))$$
$$= [(f \times A)^A(\eta^\circ_V(t_i))]_{i \in I}(\iota_i(a)) \stackrel{ind.\ hyp.}{=} [\eta^\circ_{V'}(\overline{T_\Sigma}(f)(t_i))]_{i \in I}(\iota_i(a))$$
$$= \eta^\circ_{V'}(\overline{T_\Sigma}(f)(t_i))(a) = \eta^\circ_{V'}((\overline{T_\Sigma}(f)(t_i))_{i \in I})(\iota_i(a)) = \eta^\circ_{V'}(\overline{T_\Sigma}(f)(t))(\iota_i(a)).$$

For all $e = \prod_{i\in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $t \in \overline{T_\Sigma}(V)_{e_i}$ and $a \in A_e$,

$$(f \times A)^A(\eta^\circ_V(i(t)))(a) = (\lambda(x,a).(f(x),a) \circ \eta^\circ_V(i(t)))(a)$$
$$= (\lambda(x,a).(f(x),a) \circ \eta^\circ_V(t) \circ \pi_i)(a) = ((f \times A)^A(\eta^\circ_V(t)) \circ \pi_i)(a)$$
$$\stackrel{ind.\ hyp.}{=} (\eta^\circ_{V'}(\overline{T_\Sigma}(f)(t)) \circ \pi_i)(a) = \eta^\circ_{V'}(\overline{T_\Sigma}(f)(i(t)))(a).$$

19.13 Varieties

Let $\Sigma = (S, F)$ be a constructive signature, R be a Σ-congruence on $T_\Sigma(V)$ (or an isomorphic Σ-algebra; see section 9.10), $\mathcal{A}$ be a Σ-algebra with carrier A and $g \in A^V$ (see section 9.11) 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 9.9 implies

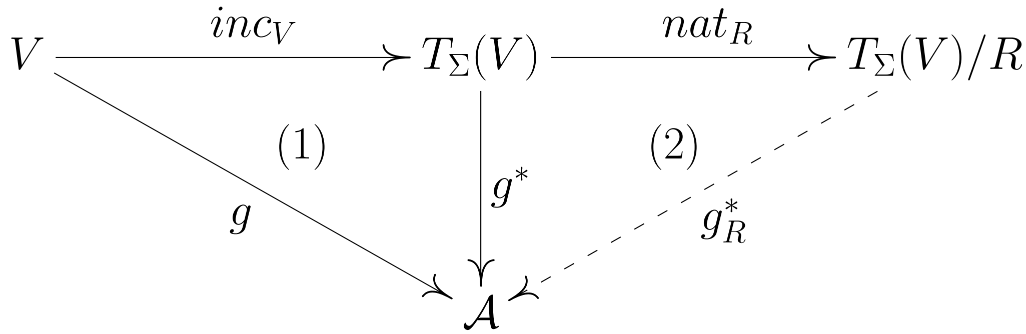
$$g^*(t) = (nat_R \circ \sigma)^*(t) = nat_R(\sigma^*(t')) = (nat_R \circ \sigma)^*(t') = g^*(t').$$

$Alg_{\Sigma,R}$, the full subcategory of all Alg_Σ whose objects satisfy R, is called a **variety**.

Hence g solves R iff $R \subseteq ker(g^*)$ iff $g_R^* : T_\Sigma(V)/R \to \mathcal{A}$ is well-defined by

$$g_R^*(nat_R(t)) = g^*(t)$$

for all $t \in T_\Sigma(V)$ iff $g^* : T_\Sigma(V) \to \mathcal{A}$ factors through $T_\Sigma(V)/R$, i.e., there is g_R^* such that $g^* = g_R^* \circ nat_R$.



Since $nat_\sim$ is epi, Lemma 9.1 (1) implies that g^* is Σ-homomorphic. Hence g_R^* is unique with (2), again because nat_R is epi.

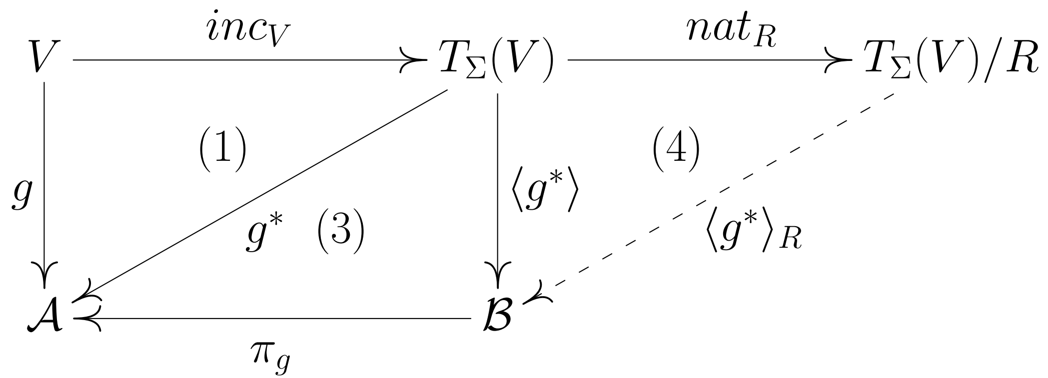
We conclude that $T_\Sigma(V)/R$ is **free in** $Alg_{\Sigma,R}$, i.e., for all $\mathcal{A} \in Alg_{\Sigma,R}$ there is a unique Σ-homomorphism from $T_\Sigma(V)/R$ to $\mathcal{A}$ with (2).

In particular, T_Σ/R is initial in $Alg_{\Sigma,R}$.

Let $\mathcal{B} = \mathcal{A}^{(A^V)}$. By equation 2.2.9,

$$\bigcap_{g \in A^V} ker(g^*) = ker(\langle g^* \rangle_{g \in A^V} : T_\Sigma(V) \to \mathcal{B}).$$

The Σ-algebra $\mathcal{B}$ becomes a $\Sigma(V)$-algebra by defining $val^{\mathcal{B}}(x)(g) = g(x)$ for all $x \in V$ and $g \in A^V$. For the definition of $\Sigma(V)$, see section 9.11.



Hence the Σ-homomorphism $\langle g^* \rangle$ is compatible with val and thus is also $\Sigma(V)$-homomorphic: For all $x \in V$ and $g' \in A^V$,

$$val^{\mathcal{B}}(\langle g^* \rangle(x))(g') = val^{\mathcal{B}}(x)(g') = g'(x) \overset{(2)}{=} (g')^*(x) \overset{(3)}{=} \pi_{g'}(\langle g^* \rangle(x)) = \langle g^* \rangle(x)(g')$$
$$= \langle g^* \rangle(val^{T_\Sigma(V)}(x))(g').$$

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

$$\Sigma(Y) = (\{state, X, Y\}, \{\delta : state \times X \to state,\ val : Y \to state\})$$

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 Σ-algebras $\mathcal{A}$ with carrier Q, $fold^{\mathcal{A}^{(Q^Y)}} = \langle g^* \rangle_{g:Y \to Q}$ (see sample algebra 9.6.3, Example 9.3 in chapter 9 and adjunction 19.3). ❏

Theorem 19.7 (Birkhoff's variety theorem; special case of [11], Theorem 6.2)

A class of Σ-algebras is a Σ-variety iff it is closed under the formation of subalgebras, homomorphic images and products. ❏

19.14 Equational theories

Let E be an S-sorted set of Σ-equations (see section 9.11).

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 (Σ, E)**-algebra** is a Σ-algebra that satisfies (all equations of) E.

$Alg_{\Sigma,E}$ denotes the full subcategory of Alg_Σ that consists of all (Σ, E)-algebras.

$RAlg_{\Sigma,E} =_{def} Alg_{\Sigma,E} \cap RAlg_\Sigma$ (see section 9.11).

The least Σ-congruence R on $T_\Sigma(V)$ such that for all $\bigwedge_{i=1}^n t_i = t_i' \Rightarrow t = t' \in E$ and $\sigma \in T_\Sigma(V)^V$,

$$\bigwedge_{i=1}^n (\sigma^*(t_i), \sigma^*(t_i')) \in R \quad \text{implies} \quad (\sigma^*(t), \sigma^*(t')) \in R, \tag{5}$$

is called the **deductive theory of** (Σ, E) and denoted by $DTh(E)$.

The notion was introduced in [118] (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

$$\begin{array}{rcl} \Phi : \mathsf{X}_{s\in S}\mathcal{P}(T_\Sigma(V)^2_s) & \to & \mathsf{X}_{s\in S}\mathcal{P}(T_\Sigma(V)^2_s) \\ R & \mapsto & (inst(R_s) \cup cong(R_s) \cup \Delta^2_{T_\Sigma(V)_s} \cup R_s^{-1} \cup R_sR_s)_{s\in S} \end{array}$$

where

$$\begin{array}{rcl} inst(R_s) & = & \{(\sigma^*(t), \sigma^*(t')) \mid \bigwedge_{i=1}^n t_i = t'_i \Rightarrow t = t' \in E_s,\ \sigma \in T_\Sigma(V)^V, \\ & & \forall\, 1 \le i \le n : (\sigma^*(t_i), \sigma^*(t'_i)) \in R\}, \\ cong(R_s) & = & \{(ft, ft') \mid f : e \to s \in F,\ (t, t') \in R_e\}. \end{array}$$

Let $\mathcal{A}$ be a (Σ, E)-algebra and R be the S-sorted set defined by $R_s = \{(t, t') \in T_\Sigma(V)^2_s \mid \mathcal{A} \models t = t'\}$ for all $s \in S$.

Since $lfp(\Phi) = DTh(E)$, fixpoint induction (see chapter 3) implies $DTh(E) \subseteq R$ if R is Φ-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 $1 \le i \le n$.

Hence Lemma 9.9 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: $(t, t') \in cong(R_s)$. Then $t = fu$ and $t' = fu'$ for some $f : e \to s \in F$ and $(u, u') \in R_e$. Hence Lemma 9.9 implies

$$g^*(t) = g^*(fu) = f^{\mathcal{A}}(g^*(u)) = f^{\mathcal{A}}(g^*(u')) = g^*(fu') = g^*(t'),$$

i.e., $(t, t') \in R_s$.

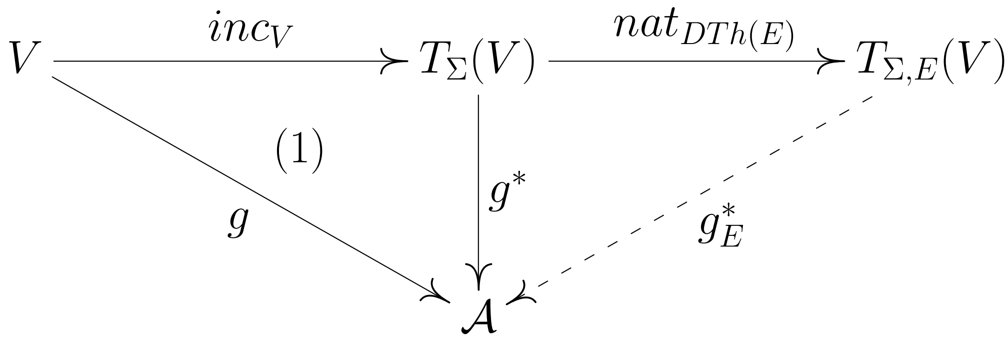
Case 3: $t = t'$. Then $(t, t') \in R_s$ because R_s is reflexive.

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 Φ-closed. ❏

$T_{\Sigma,E}(V) =_{def} T_\Sigma(V)/DTh(E)$ is a **free** (Σ, E)**-algebra over** V, i.e., for all (Σ, E)-algebras $\mathcal{A}$ with carrier A and $g \in A^V$ there is a unique Σ-homomorphism $g^*_E : T_{\Sigma,E}(V) \to \mathcal{A}$ with $g^*_E \circ nat_{DTh(E)} = g^*$.



Proof. Since $DTh(E)$ is a Σ-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 $g = nat \circ \sigma$ for some $\sigma : V \to T_\Sigma(V)$ and the natural map $nat : T_\Sigma(V) \to T_{\Sigma,E}(V)$. Since $T_{\Sigma,E}(V)$ is a Σ-quotient of $T_\Sigma(V)$, nat is Σ-homomorphic. Hence by Lemma 9.9,

$$nat(\sigma^*(t_i)) = (nat \circ \sigma)^*(t_i) = g^*(t_i) = g^*(t'_i) = (nat \circ \sigma)^*(t'_i) = nat(\sigma^*(t'_i)),$$

i.e., $(\sigma^*(t_i), \sigma^*(t_i')) \in DTh(E)$. Therefore, $(\sigma^*(t), \sigma^*(t')) \in inst(DTh(E)) \subseteq \Phi(DTh(E))$ (see above).

Since $DTh(E) = lfp(\Phi)$ is Φ-closed, $(\sigma^*(t), \sigma^*(t')) \in DTh(E)$. Hence

$$g^*(t) = (nat \circ \sigma)^*(t) = nat(\sigma^*(t)) = nat(\sigma^*(t')) = (nat \circ \sigma)^*(t') = g^*(t').$$

We conclude that $T_{\Sigma,E}(V)$ satisfies e.

Let $\mathcal{A}$ be a Σ-algebra with carrier A that satisfies E and $g \in A^V$. Suppose that the kernel of $g^* : T_\Sigma \to A$ is Φ-closed.

Since $DTh(E)$ is the least Φ-closed subset of $T_\Sigma(V)^2$,

$$DTh(E) \text{ is a subset of } ker(g^*). \tag{8}$$

Hence $g_E^* : T_{\Sigma,E} \to A$ is well-defined by $g_E^* \circ nat_{DTh(E)} = g^*$.

Since g^* is Σ-homomorphic and $nat_{DTh(E)}$ is epi in Alg_Σ, Lemma 9.1 (2) implies that g_E^* is Σ-homomorphic. Let $h : T_{\Sigma,E} \to A$ be any Σ-homomorphism with $h \circ nat_{DTh(E)} = g^*$. Hence $h \circ nat_{DTh(E)} \circ inc_V = g^* \circ inc_V$. Since g^* is the only Σ-homomorphism from T_Σ to $\mathcal{A}$ with $g^* \circ inc_V = g$, $h \circ nat_{DTh(E)} = g_E^* \circ nat_{DTh(E)}$. Since $nat_{DTh(E)}$ is epi, $h = g_E^*$.

It remains to show that $R = ker(g^*)$ is Φ-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 9.9,

$$g^*(t) = g^*(\sigma^*(u)) = (g^* \circ \sigma)^*(t) = (g^* \circ \sigma)^*(t') = g^*(\sigma^*(u')) = g^*(t'),$$

i.e., $(t, t') \in R_s$.

Case 2: $t = fu$ and $t' = fu'$ for some $f : e \to s \in F$ and $(u, u') \in R_e$. Hence by Lemma 9.9,

$$g^*(t) = g^*(fu) = f^{\mathcal{A}}(g^*(u)) = f^{\mathcal{A}}(g^*(u')) = g^*(fu') = g^*(t'),$$

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 Φ-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 (Σ, 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 nat be the natural map from $T_\Sigma(V)$ to $T_{\Sigma,E}(V)$. Then by Lemma 9.9,

$$\begin{aligned} & nat(t) = nat(id(t)) = nat(inc_V^*(t)) = (nat \circ inc_V)^*(t) = (nat \circ inc_V)^*(t') \\ & = nat(inc_V^*(t')) = nat(id(t')) = nat(t'), \end{aligned}$$

i.e., $(t,t') \in DTh(E)$. ❑

Examples

1. Let $\Sigma = Mon$, $x, y, z \in V$ and

$$E \;=\; \{mul(x, mul(y,z)) = mul(mul(x,y),z),\; mul(x, one) = x,\; mul(one, x) = x\},$$

Then $T_{\Sigma,E}(V) \cong V^*$.

2. Let $\Sigma = List(X)$, $s \in V_{list}$ and

$$\begin{array}{rcl} E & = & \{cons(x, cons(y, s)) = cons(y, cons(x, s)) \mid x, y \in X\} \cup \\ & & \{cons(x, cons(x, s)) = cons(x, s) \mid x \in X\}. \end{array}$$

Then $T_{\Sigma,E} \cong \mathcal{P}_\omega(X)$ is initial in $Alg_{\Sigma,E}$ where $\alpha^{\mathcal{P}_\omega(X)} = \emptyset$ and for all $x \in X$ and finite subsets S of X, $cons^{\mathcal{P}_\omega(X)}(x, S) = S \cup \{x\}$.

3. Let $\Sigma = ARRAY$, $s \in V_{list}$ 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 (Σ, E), a notion introduced in [118, 119] to remind of the fact that the equations of $ITh(E)$ can be proved by induction on T_Σ.

Soundness and completeness of $ITh(E)$ w.r.t. $RAlg_{\Sigma,E}$ and $T_{\Sigma,E}$

For all $e = (t, t') \in T_\Sigma(V)^2$,

$$e \in ITh(E) \quad \text{iff} \quad RAlg_{\Sigma,E} \models t = t' \quad \text{iff} \quad T_{\Sigma,E} \models t = t'.$$

Proof. Let $e = (t, t') \in ITh(E)$, $\mathcal{A}$ be a reachable Σ-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 9.9 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 (Σ, E)-algebras satisfy $t = t'$. By (7), $T_{\Sigma,E}$ satisfies E. Since T_Σ is initial in Alg_Σ and $nat = nat_{\sim_E} : T_\Sigma \to T_{\Sigma,E}$ is Σ-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 9.9,

$$\begin{aligned} & nat(\sigma^*(t)) = fold^{T_{\Sigma,E}}(\sigma^*(t)) = (fold^{T_{\Sigma,E}} \circ \sigma)^*(t) = (fold^{T_{\Sigma,E}} \circ \sigma)^*(t') \\ & = fold^{T_{\Sigma,E}}(\sigma^*(t')) = nat(\sigma^*(t')), \end{aligned}$$

i.e., $(\sigma^*(t), \sigma^*(t')) \in DTh(E)$ and thus $(t, t') \in ITh(E)$. ❏

19.15 Coterm functors

Let $\Sigma = (S, D)$ be a destructive polynomial signature, U_S be the forgetful functor from Alg_Σ to Set^S, $C, C' \in Set^S$ and $\mathcal{A}$ be a Σ-algebra (see chapter 9).

$\epsilon_C =_{def} root =_{def} \lambda t.t(\epsilon) : DT_\Sigma(C) \to C$.

$$\frac{U_S(\mathcal{A}) \xrightarrow{g} C}{\mathcal{A} \xrightarrow{g^\#} DT_\Sigma(C)} \qquad\qquad \frac{\mathcal{A} \xrightarrow{h} DT_\Sigma(C)}{U_S(\mathcal{A}) \xrightarrow{h^*=\epsilon_C \circ h} C}$$

The functor

$$\begin{array}{rcl} DT_\Sigma : Set^S & \to & Alg_\Sigma \\ C & \mapsto & DT_\Sigma(C) \\ f : C' \to C & \mapsto & (f \circ \epsilon_{C'})^\# : DT_\Sigma(C') \to DT_\Sigma(C) \end{array}$$

is right adjoint to U_S. Proof of the functor property: Let $g : C'' \to C'$. 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 9.12,

$$DT_\Sigma(f \circ g) = (f \circ g \circ \epsilon_{C''})^\# = (f \circ \epsilon_{C'})^\# \circ (g \circ \epsilon_{C''})^\# = DT_\Sigma(f) \circ DT_\Sigma(g).$$

For all Σ-algebras $\mathcal{A}$ with carrier A, the unit $\eta_A = id_A^\# : \mathcal{A} \to DT_\Sigma(A)$ 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 Σ.

Since $C \in Set^S$ with $C_s = 1$ for all $s \in S$ is final in Set^S and right adjoints preserve final objects, DT_Σ is final in Alg_Σ, which also follows from the definition of the coextension

$$unfold^{\mathcal{A}} : \mathcal{A} \to DT_\Sigma$$

(see section 9.16).

$DT_\Sigma(C)$ represents $F_C =_{def} Set^S(U_S(\_), C)$ (see chapter 5) because for all $C \in Set^S$, the contravariant functors F_C and $Alg_\Sigma(\_, DT_\Sigma(C))$ are naturally equivalent, i.e., for all Σ-algebras $\mathcal{A}$, the set $F_C(\mathcal{A})$ of colorings of (the carrier of) $\mathcal{A}$ by C and the set $Alg_\Sigma(\mathcal{A}, DT_\Sigma(C))$ of Σ-homomorphisms are isomorphic. Moreover, by Corollary 5.2 (8), this applies as well to isomorphic representations of $DT_\Sigma(C)$.

Let $\Sigma(C)$ be defined as in section 9.17. 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 $DT_\Sigma \cong BF_X = \_^{X^*}$ (see section 19.5) and $\Sigma(Y) = DAut(X,Y)$. Hence $BF_X(Y)$ is final in $Alg_{DAut(X,Y)}$ and for all $DAut(X,Y)$-algebras $\mathcal{A}$ with carrier A, $(\beta^{\mathcal{A}})^\# = unfold^{\mathcal{A}} = BF_X(\beta^{\mathcal{A}}) \circ id_A^\# = (\beta^{\mathcal{A}})^{X^*} \circ id_A^\#$.

In particular, since $\Sigma(2) = DAut(X,2) = Acc(X)$, $BF_X(2) \cong Pow(X)$ is final in $Alg_{Acc(X)}$ (see sample final algebra 9.18.10).

19.16 Covarieties

Let $\Sigma = (S,F)$ be a destructive signature, I be a Σ-invariant of $DT_\Sigma(C)$ (or an isomorphic Σ-algebra; see section 9.9), $\mathcal{A}$ be a Σ-algebra with carrier A and $g \in C^A$ (see section 9.16) such that for all $t \in I$ and $\sigma \in C^{DT_\Sigma(C)}$, $\sigma^\#(t) \in I$.

g **solves** I in $\mathcal{A}$ 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.

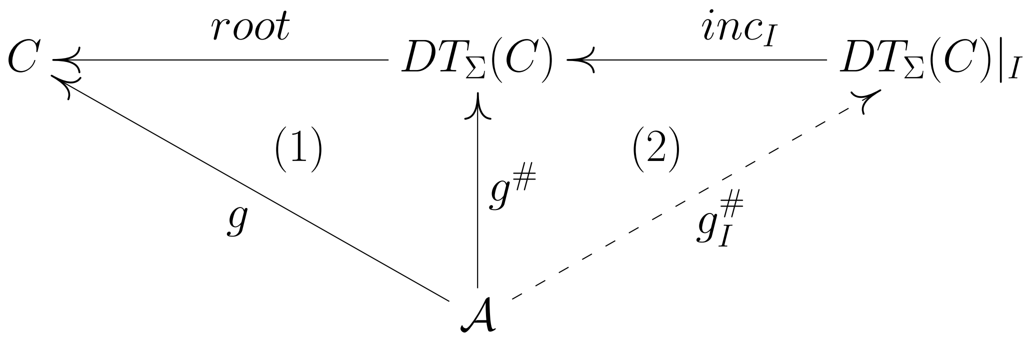
$DT_\Sigma(C)|_I$ satisfies I: Let $g \in C^I$. Then $\sigma \circ inc_I = g$ for some $\sigma \in C^{DT_\Sigma(C)}$. Hence for all $t \in I$, Lemma 9.16 implies $g^\#(t) = (\sigma \circ inc_I)^\#(t) = \sigma^\#(t) \in I$.

$Alg_{\Sigma,I}$, the full subcategory of all Alg_Σ whose objects satisfy I, is called a **covariety**.

Hence g solves I iff $img(g^\#) \subseteq I$ iff $g_I^\# : \mathcal{A} \to I$ is well-defined by

$$g_I^\#(a) = g^\#(a)$$

for all $a \in A$ iff $g^\# : \mathcal{A} \to DT_\Sigma(C)$ factors through I, i.e., there is $g_I^\#$ such that $g^\# = inc_I \circ g_I^\#$.



Since inc_I is mono, Lemma 9.1 (2) implies that $g^\#$ is Σ-homomorphic. Hence $g_I^\#$ is unique with (2), again because inc_I is mono.

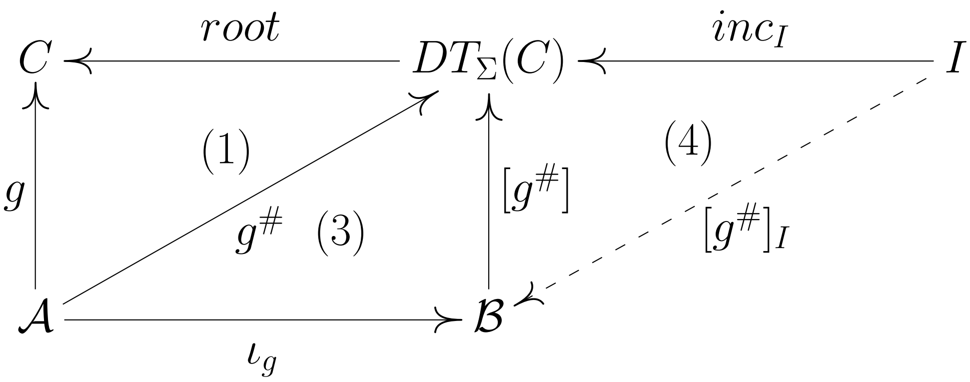
We conclude that $DT_\Sigma(C)|_I$ is **cofree in** $Alg_{\Sigma,I}$, i.e., for all $\mathcal{A} \in Alg_{\Sigma,I}$ there is a unique Σ-homomorphism $g_I^\# : \mathcal{A} \to DT_\Sigma(C)|_I$ with (2).

In particular, $DT_\Sigma|_I$ is final in $Alg_{\Sigma,I}$.

Let $\mathcal{B} = \mathcal{A} \times C^A$. By equation 2.5.20,

$$\bigcup_{g \in C^A} img(g^\#) = img([g^\#]_{g \in C^A} : \mathcal{B} \to DT_\Sigma(C)).$$

The Σ-algebra $\mathcal{B}$ becomes a $\Sigma(C)$-algebra by defining $col^{\mathcal{B}}(a, g) = g(a)$ for all $a \in A$ and $g \in C^A$. For the definition of $\Sigma(C)$, see section 9.16.



Hence the Σ-homomorphism $[g^\#]$ is compatible with *col* and thus also $\Sigma(C)$-homomorphic: For all $a \in A$ and $g' \in C^A$,

$$[g^\#](col^{\mathcal{B}}(a, g')) = col^{\mathcal{B}}(a, g') = g'(a) = root((g')^\#(a)) = root([g^\#](\iota_{g'}(a)))$$
$$= col^{DT_\Sigma(C)}([g^\#](\iota_{g'}(a))) = col^{DT_\Sigma(C)}([g^\#](a, g')).$$

Therefore, $unfold^{\mathcal{B}} = [g^\#]$.

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

$$\Sigma(Y) = (\{state, X, Y\}, \{\delta : Q \to Q^X,\ col : state \to Y\})$$

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 Σ-algebras $\mathcal{A}$ with carrier Q, $unfold^{\mathcal{A}\times Y^Q} = [g^\#]_{g:Q\to Y}$ (see sample algebra 9.6.24, Example 9.4 and section 19.5). ❏

Theorem 19.8 (Birkhoff's covariety theorem; special case of [11], Theorem 6.15)

A class of Σ-algebras is a Σ-covariety iff it is closed under the formation of subalgebras, homomorphic images and coproducts. ❏

19.17 Coequational theories

Let E be an S-sorted set of Σ-coequations (see section 9.16).

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 (Σ, E)**-algebra** is a Σ-algebra that satisfies (all coequations of) E.

$Alg_{\Sigma,E}$ denotes the full subcategory of Alg_Σ that consists of all (Σ, E)-algebras.

$OAlg_{\Sigma,E} =_{def} Alg_{\Sigma,E} \cap OAlg_\Sigma$ (see section 9.16).

The greatest Σ-invariant P of $DT_\Sigma(C)$ such that for all $ex(t) \Rightarrow \bigvee_{i=1}^n ex(t_i) \in E$ and $\sigma \in C^{DT_\Sigma(C)}$,

$$\sigma^\#(t) \in P \quad \text{implies} \quad \bigvee_{i=1}^{n} \sigma^\#(t_i) \in P \tag{5}$$

is called the **deductive theory** of (Σ, E) and denoted by $DTh(E)$.

Completeness of $DTh(E)$ w.r.t. $Alg_{\Sigma,E}$

For all $t \in DT_\Sigma(C)$, $Alg_{\Sigma,E} \not\models \neg ex(t)$ implies $t \in DTh(E)$. (6)

Proof. By definition, $DTh(E)$ coincides with the greatest fixpoint of

$$\begin{aligned}\Phi : \times_{s\in S}\mathcal{P}(DT_\Sigma(C)_s) &\to \times_{s\in S}\mathcal{P}(DT_\Sigma(C)_s) \\ P &\mapsto (coinst(P_s)\cap inv(P_s))_{s\in S}\end{aligned}$$

where

$$\begin{aligned}coinst(P_s) &= \{t\in DT_\Sigma(C)_s \mid \forall\ ex(u)\Rightarrow \textstyle\bigvee_{i=1}^n ex(u_i)\in E_s,\ \sigma\in C^{DT_\Sigma(C)}: \\ &\qquad \sigma^\#(t)=u\Rightarrow \exists\, 1\le i\le n,\ t'\in P:\sigma^\#(t')=u_i\}, \\ inv(P_s) &= \{t\in DT_\Sigma(C)_s \mid \forall\ f:s\to e\in F: f^{DT_\Sigma(C)}(t)\in P_e\}.\end{aligned}$$

Let P be the S-sorted set defined by $P_s=\{t\in DT_\Sigma(C)_s \mid Alg_{\Sigma,E}\not\models \neg ex(t)\}$ for all $s\in S$.

Since $gfp(\Phi)=DTh(E)$, fixpoint coinduction (see chapter 3) implies $P\subseteq DTh(E)$ if P is Φ-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 $Alg_{\Sigma,E}\not\models\neg ex(t)$, i.e., $g^\#(a)=t$ for some (Σ,E)-algebra $\mathcal{A}$ with carrier A, $g\in C^A$ and $a\in A_s$. Hence by Lemma 9.16,

$$(\sigma\circ g^\#)^\#(a)=\sigma^\#(g^\#(a))=\sigma^\#(t)=u$$

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 $t \in coinst(P_s)$.

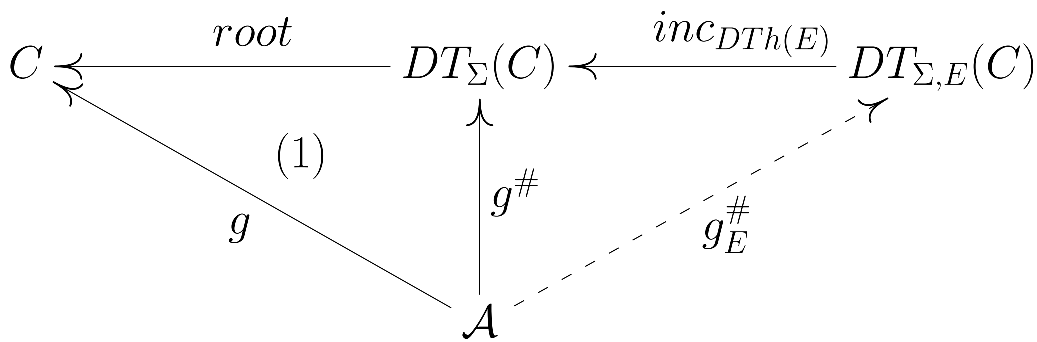
Moreover, for all $f : s \to e \in F$,

$$f^{DT_\Sigma(C)}(t) = f^{DT_\Sigma(C)}(g^{\#}(a)) = g^{\#}(f^{\mathcal{A}}(a)),$$

i.e., $f^{DT_\Sigma(C)}(t) \in P_e$. Hence $t \in inv(P_s)$.

Therefore, $t \in \Phi(P_s) = coinst(P_s) \cap inv(P_s)$. We conclude that P is Φ-dense. ❑

$DT_{\Sigma,E}(C) =_{def} DT_\Sigma(C)|_{DTh(E)}$ is a **cofree** (Σ, E)**-algebra over** C, i.e., for all (Σ, E)-algebras $\mathcal{A}$ with carrier A and $g \in C^A$ there is a unique Σ-homomorphism $g_E^{\#} : \mathcal{A} \to DT_{\Sigma,E}(C)$ with $inc_{DTh(E)} \circ g_E^{\#} = g^{\#}$.



In particular, $DT_{\Sigma,E}$ is final in $Alg\Sigma, E$.

Proof. Since $DTh(E)$ is a Σ-invariant, $DT_{\Sigma,E}$ is well-defined. Next we show that $DT_{\Sigma,E}(C)$ is a (Σ, 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 $inc : DTh(E) \to DT_\Sigma(C)$. Since $DT_{\Sigma,E}(C)$ is a Σ-subalgebra of $DT_\Sigma(C)$, inc is Σ-homomorphic. Hence by Lemma 9.16,

$$u = g^{\#}(t) = (\sigma \circ inc)^{\#}(t) = \sigma^{\#}(inc(t)) = \sigma^{\#}(t).$$

Since $t \in DTh(E)$ and $DTh(E) = gfp(\Phi)$ is Φ-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 9.16,

$$g^{\#}(t') = (\sigma \circ inc)^{\#}(t') = \sigma^{\#}(inc(t')) = \sigma^{\#}(t') = u_i.$$

We conclude that $DT_{\Sigma,E}(C)$ satisfies e.

Let $\mathcal{A}$ be a Σ-algebra with carrier A that satisfies E and $g \in C^A$. Suppose that the image of $g^{\#} : A \to DT_\Sigma$ is Φ-dense.

Since $DTh(E)$ is the greatest Φ-dense subset of $DT_\Sigma(C)$,

$$img(g^{\#}) \text{ is a subset of } DTh(E). \quad (8)$$

Hence $g_E^\# : A \to DTh(E)$ is well-defined by $inc_{DTh(E)} \circ g_E^\# = g^\#$.

Since $g^\#$ is Σ-homomorphic and $inc_{DTh(E)}$ is mono in Alg_Σ, Lemma 9.1 (2) implies that $g_E^\#$ is Σ-homomorphic. Let $h : A \to DTh(E)$ be any Σ-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 Σ-homomorphism from $\mathcal{A}$ to $DT_\Sigma(C)$ with $root \circ g^\# = g$, $inc_{DTh(E)} \circ h = inc_{DTh(E)} \circ g_E^\#$. Since $inc_{DTh(E)}$ is mono, $h = g_E^\#$.

It remains to show that $P = img(g^\#)$ is Φ-dense. 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 $t = g^\#(a)$ for some $a \in A$. Hence by Lemma 9.16,

$$(\sigma \circ g^\#)^\#(a) = \sigma^\#(g^\#(a)) = \sigma^\#(t) = u$$

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 $t \in coinst(P_s)$.

Moreover, for all $f : s \to e \in F$,

$$f^{DT_\Sigma(C)}(t) = f^{DT_\Sigma(C)}(unfold^\mathcal{A}(a)) = unfold^\mathcal{A}(f^\mathcal{A}(a)),$$

i.e., $f^{DT_\Sigma(C)}(t) \in P_e$. Hence $t \in inv(P_s)$.

Therefore, $t \in \Phi(P_s) = coinst(P_s) \cap inv(P_s)$. We conclude that P is Φ-dense. ❑

Soundness and completeness of $DTh(E)$ w.r.t. $Alg_{\Sigma,E}$ and $DT_{\Sigma,E}(C)$

For all $t \in DT_\Sigma(C)$,

$$Alg_{\Sigma,E} \not\models \neg ex(t) \ \text{ iff } \ t \in DTh(E) \ \text{ iff } \ DT_{\Sigma,E}(C) \not\models \neg ex(t).$$

Proof. Suppose that $Alg_{\Sigma,E}$ does not satisfy $\neg ex(t)$. 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 9.16, $(root \circ inc)^\#(t) = root^\#(inc(t)) = id(inc(t)) = t$. Hence $DT_{\Sigma,E}(C)$ does not satisfy $\neg ex(t)$.

Suppose that $DT_{\Sigma,E}(C)$ does not satisfy $\neg ex(t)$. By (7), $DT_{\Sigma,E}(C)$ satisfies E. Hence $Alg_{\Sigma,E}$ does not satisfy $\neg ex(t)$. ❑

Examples

1. Let $\Sigma = (\{state, 1\}, \{f : state \to state\})$ and $\mathcal{A}$ be a Σ-algebra with carrier A and $\delta = f^\mathcal{A}$.

Then for all $a \in A$ and $n \in \mathbb{N}$,

$$g^\#(a)(f^n) = (f^{DT_\Sigma(C)})^n(g^\#(a))(\epsilon) = g^\#(\delta^n(a))(\epsilon). \tag{9}$$

Let $V_{state} = \{(), x\}$, $t = ()\{f \to t'\}$, $t' = x\{f \to t'\}$, $t_1 = ()\{f \to t\}$ and $t_2 = x\{f \to t'\}$.

1.1 ([61], 6.1) Let $\mathcal{K}$ be the category of Σ-algebras $\mathcal{A}$ with carrier A and $\delta = f^{\mathcal{A}}$ such that for all $a \in A$ there is $n > 0$ with $\delta^n(a) = a$.

$\mathcal{K} = Alg_{\Sigma,\{\neg ex(t)\}}$.

Proof. Let $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma,\{\neg ex(t)\}}$. Then $a = \delta^n(a)$ and $g^\#(a) = t$ for some $n > 0$, $g \in V^A$ and $a \in A$. Hence

$$() = t(\epsilon) = g^\#(\delta^n(a))(\epsilon) \stackrel{(9)}{=} g^\#(a)(f^n) = t(f^n) = x. \ \lightning$$

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, $\mathcal{A} \notin Alg_{\Sigma,\{\neg ex(t)\}}$.

1.2 ([61], 6.2) Let $\mathcal{K}'$ be the category of Σ-algebras $\mathcal{A}$ with carrier A and $\delta = f^{\mathcal{A}}$ such that δ 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 $\neg ex(t)$ 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 $\neg ex(t)$, i.e., $g^\#(a) = t$ for some $g \in V^A$ and $a \in A$. Since δ is surjective, there is $b \in A$ such that $\delta(b) = a$. Hence

$$g^\#(b) = g(b)\{f \to g^\#(\delta(b))\} = g(b)\{f \to g^\#(a)\} = g(b)\{f \to t\}$$

and thus $g^\#(b) = t_1$ or $g^\#(b) = t_2$ because $g(b) \in \{(), x\}$.

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 $\mathcal{K} = Alg_{\Sigma,\{\neg ex(t)\}}$). 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 δ is surjective, i.e., $\mathcal{A} \in \mathcal{K}'$.

1.3 ([11], Ex. 4.15 (a)) Let $t = ()\{f \to x\{f \to t\}\}$ and $\mathcal{K}$ be the category of Σ-algebras $\mathcal{A}$ with carrier A and $\delta = f^{\mathcal{A}}$ such that for all $a \in A$ there are $k, n \in \mathbb{N}$ such that $\delta^k(a) = \delta^{k+2n+1}(a)$.

$\mathcal{K} = Alg_{\Sigma,\{\neg ex(t)\}}$.

Proof. Let $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma,\{\neg ex(t)\}}$. 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

$$t(f^k) = g^{\#}(a)(f^k) \overset{(9)}{=} g^{\#}(\delta^k(a))(\epsilon) = g^{\#}(\delta^{k+2n+1}(a))(\epsilon) \overset{(9)}{=} g^{\#}(a)(f^{k+2n+1}) = t(f^{k+2n+1}). \lightning$$

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 $g = \lambda a'.\mathit{if}\ \exists\, n \in \mathbb{N} : a' = \delta^{2n}(a)\ \mathit{then}\ ()\ \mathit{else}\ x$.

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 $n \in \mathbb{N}$,

$$g^{\#}(a)(f^{2n}) \overset{(9)}{=} g^{\#}(\delta^{2n}(a))(\epsilon) = g(\delta^{2n}(a)) = * = t(f^{2n}),$$
$$g^{\#}(a)(f^{2n+1}) \overset{(9)}{=} g^{\#}(\delta^{2n+1}(a))(\epsilon) = g(\delta^{2n+1}(a)) = x = t(f^{2n+1}),$$

i.e., $g^{\#}(a) = t$. Therefore, $\mathcal{A} \notin Alg_{\Sigma,\{\neg ex(t)\}}$.

2. ([7], 2.5; [11], Ex. 4.14; [163], Exs. 4.3 and 4.6) Let

$$\Sigma = (\{state, \{1,2\}\}, \{f : state \to 1 + (state \times state)\})$$

and $V_{state} = \{(), x\}$.

2.1 Let $t = ()\{f \to 1(\epsilon)\}$ and $\mathcal{K}$ be the category of Σ-algebras $\mathcal{A}$ with carrier A and $\delta = f^{\mathcal{A}}$ such that for all $a \in A$, $\delta(a) = \iota_2(b,c)$ for some $b, c \in A$.

$\mathcal{K} = Alg_{\Sigma,\{\neg ex(t)\}}$.

Proof. Let $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma,\{\neg ex(t)\}}$. 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

$$\begin{aligned}&()\{1 \to g^\#(b), 2 \to g^\#(c)\} = g^\#(b,c) = g^\#(\delta(a)) = f^{DT_\Sigma(C)}(g^\#(a))\\ &= f^{DT_\Sigma(C)}(t) = 1(\epsilon). \lightning\end{aligned}$$

Conversely, let $\mathcal{A} \in Alg_\Sigma \setminus \mathcal{K}$ and $g = \lambda a.\epsilon$. Then $\delta(a) = \epsilon$ for some $a \in A$. Hence

$$g^\#(a) = g(a)\{f \to 1(g^\#(\epsilon))\} = ()\{f \to 1(\epsilon)\} = t$$

and thus $\mathcal{A} \notin Alg_{\Sigma,\{\neg ex(t)\}}$.

2.2 Let $t = ()\{f \to 2(()\{1 \to t, 2 \to t\})\}$, $t' = x\{f \to 2(()\{1 \to t, 2 \to t\})\}$, $\mathcal{K}$ be the category of all Σ-algebras $(A, \{\delta\})$ such that for all $a \in A$,

$$\delta(b) = \epsilon \text{ for some } b \in \langle a \rangle, \tag{10}$$

and $\mathcal{K}'$ be the category of Σ-algebras $\mathcal{A}$ with carrier A and $\delta = f^{\mathcal{A}}$ such that for all $a \in A$, (10) holds true or $\delta(a) = \iota_2(b)$ implies $a \in \langle b \rangle$.

$\mathcal{K} = Alg_{\Sigma,\{\neg ex(t)\}}$.

Proof. Let $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma,\{\neg ex(t)\}}$. Then $g^\#(a) = t$ and (10) holds true for some $g \in V^A$ and $a \in A$. Hence $g^\#(b) = ()\{f \to 2(u)\}$ for some subcoterm $u \neq *$ of t. Therefore,

$$() = g^\#(\epsilon) = g^\#(\epsilon) = g^\#(\delta(b)) = f^{DT_\Sigma(C)}(g^\#(b)) = f^{DT_\Sigma(C)}(()\{f \to 2(u)\}) = 2(u). \ \lightning$$

Conversely, let $\mathcal{A} \in Alg_\Sigma \setminus \mathcal{K}$ 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 $w \in def(g^\#(a))$ there is $w' \in \{f, 1, 2\}^*$ such that $w \in \{f * 1w', f * 2w', f * w'\}$.

If $w = f()1w'$, then

$$g^{\#}(a)(w) = g^{\#}(b)(w') \stackrel{ind.\ hyp.}{=} t(w') = t(w).$$

If $w = f*2w'$, then $g^{\#}(a)(w) = g^{\#}(c)(w') \stackrel{ind.\ hyp.}{=} t(w') = t(w)$. If $w = f*w'$, then both $g^{\#}(a)(w)$ and $t(w)$ are undefined. Hence $g^{\#}(a) = t$ and thus $\mathcal{A} \notin Alg_{\Sigma,\{\neg ex(t)\}}$.

$\mathcal{K}' = Alg_{\Sigma,\{ex(t')\Rightarrow False\}}$.

Proof. Let $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma,\{\neg ex(t)\}}$.

2.3 Let $\mathcal{K}$ be the category of all Σ-algebras $(A, \{\delta\})$ such that for all $a, b, c \in A$,

$$\delta(a) = \iota_2(b,c) \quad \Rightarrow \quad \delta(b) \neq \epsilon \vee \delta(c) \neq \epsilon, \tag{11}$$

and $t = ()\{f \to 2(()\{1 \to ()\{f \to 1(\epsilon)\}, 2 \to ()\{f \to 1(\epsilon)\})\}$.

$\mathcal{K} = Alg_{\Sigma,\{\neg ex(t)\}}$.

Proof. Let $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma,\{\neg ex(t)\}}$. Then $g^{\#}(a) = t$ for some $g \in V^A$ and $a \in A$. Hence

$$\begin{aligned} g^{\#}(\delta(a)) &= f^{DT_\Sigma(C)}(g^{\#}(a)) = f^{DT_\Sigma(C)}(t) = \lambda w.t(f*w) \\ &= ()\{1 \to ()\{f \to 1(\epsilon)\}, 2 \to ()\{f \to 1(\epsilon)\}\} \end{aligned} \tag{12}$$

and thus $\delta(a) = \iota_2(b,c)$ for some $b, c \in A$.

Therefore,

$$\begin{aligned}&(\,)\{1 \to g^{\#}(b), 2 \to g^{\#}(c)\} = g^{\#}(b,c) = g^{\#}(\delta(a))\\ &\overset{(12)}{=} (\,)\{1 \to (\,)\{f \to 1(\epsilon)\}, 2 \to (\,)\{f \to 1(\epsilon)\}\}.\end{aligned} \tag{13}$$

Moreover, by (11) and w.l.o.g., $\delta(b) = \iota_2(d,e)$ for some $d, e \in A$.

By (13), $g^{\#}(b) = (\,)\{f \to 1(\epsilon)\}$. Hence

$$\begin{aligned}&(\,)\{1 \to g^{\#}(d), 2 \to g^{\#}(e)\} = g^{\#}(d,e) = g^{\#}(\delta(b)) = f^{DT_\Sigma(C)}(g^{\#}(b))\\ &= \lambda w.g^{\#}(b)(f * w) = \lambda w.\mathit{if}\ w = \epsilon\ \mathit{then}\ *\ \mathit{else}\ (\,).\ \lightning\end{aligned}$$

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

$$\begin{aligned}&g^{\#}(a) = g(a)\{f \to 2((\,)\{1 \to g^{\#}(b), 2 \to g^{\#}(c)\})\}\\ &= (\,)\{f \to 2((\,)\{1 \to g(b)\{f \to 1(\epsilon)\}, 2 \to g(c)\{f \to 1(\epsilon)\}\})\}\\ &= (\,)\{f \to 2((\,)\{1 \to (\,)\{f \to 1(\epsilon)\}, 2 \to (\,)\{f \to 1(\epsilon)\}\})\} = t\end{aligned}$$

and thus $\mathcal{A} \notin Alg_{\Sigma,\{\neg ex(t)\}}$.

2.4 Let $\mathcal{K}$ be the category of all Σ-algebras A such that for all $a, b, c \in A$,

$$\delta(a) = \iota_2(b,c) \wedge \delta(b) = \epsilon = \delta(c) \quad \Rightarrow \quad b = c, \tag{14}$$

and $t = (\,)\{f \to 2((\,)\{1 \to (\,)\{f \to 1(\epsilon)\}, 2 \to x\{f \to 1(\epsilon)\}\})\}$.

$\mathcal{K} = Alg_{\Sigma,\{\neg ex(t)\}}$.

Proof. Let $\mathcal{A} \in \mathcal{K} \setminus Alg_{\Sigma,\{\neg ex(t)\}}$. Then $g^\#(a) = t$ for some $g \in V^A$ and $a \in A$. Hence

$$\begin{aligned} &g^\#(\delta(a)) = f^{DT_\Sigma(C)}(g^\#(a)) = f^{DT_\Sigma(C)}(t) = \lambda w.t(f * w) \\ &= (\,)\{1 \to (\,)\{f \to 1(\epsilon)\}, 2 \to x\{f \to 1(\epsilon)\}\} \end{aligned} \tag{15}$$

and thus $\delta(a) = \iota_2(b, c)$ for some $b, c \in A$. Therefore,

$$\begin{aligned} &(\,)\{\pi_1 \to g^\#(b), \pi_2 \to g^\#(c)\} = g^\#(b, c) = g^\#(\delta(a)) \\ &\overset{(15)}{=} (\,)\{1 \to (\,)\{f \to 1(\epsilon)\}, 2 \to x\{f \to 1(\epsilon)\}\} \end{aligned} \tag{16}$$

and thus $\delta(b) = \epsilon = \delta(c)$. By (14), $b = c$. Hence

$$(\,)\{f \to 1(\epsilon)\} \overset{(16)}{=} g^\#(b) = g^\#(c) \overset{(16)}{=} x\{f \to 1(\epsilon)\}. \;\lightning$$

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

$$
\begin{aligned}
g^{\#}(a) &= g(a)\{f \to 2(()\{1 \to g^{\#}(b), 2 \to g^{\#}(c)\})\} \\
&= ()\{f \to 2(()\{1 \to g(b)\{f \to 1(\epsilon)\}, 2 \to g(c)\{f \to 1(\epsilon)\}\})\} \\
&= ()\{f \to 2(()\{1 \to ()\{f \to 1(\epsilon)\}, 2 \to x\{f \to 1(\epsilon)\}\})\} = t
\end{aligned}
$$

and thus $\mathcal{A} \notin Alg_{\Sigma,\{\neg ex(t)\}}$.

3. (See [7], 2.6; sample final algebra 9.18.6) 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 $\Sigma = coList(X)$, $t = ()\{split \to 1(x)\}$ for some $x \in X$ and $E = \{\neg ex(t)\}$. 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 $\Sigma = coList(X)$, $t = ()\{split \to 2(()\{1 \to x, 2 \to t\})\}$ for some $x \in X$ and $E = \{\neg ex(t)\}$. 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 (Σ, E).

Soundness and completeness of $CTh(E)$ w.r.t. $OAlg_{\Sigma,E}$ and $DT_{\Sigma,E}$

For all $t \in DT_\Sigma(C)$,

$$t \in CTh(E) \text{ iff } DT_{\Sigma,E} \not\models \neg ex(t) \text{ iff } OAlg_{\Sigma,E} \not\models \neg ex(t).$$

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 $\neg ex(t)$.

Let $DT_{\Sigma,E} \not\models \neg ex(t)$. Since $unfold^{DT_{\Sigma,E}} = inc : DTh(E) \cap DT_\Sigma \to DT_\Sigma$, $DT_{\Sigma,E}$ is observable. By (7), $DT_{\Sigma,E}$ satisfies E. Hence $OAlg_{\Sigma,E} \not\models \neg ex(t)$.

Suppose that some observable (Σ, E)-algebra $\mathcal{A}$ with carrier A does not satisfy $\neg ex(t)$. 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 9.16,

$$\sigma^\#(unfold^{\mathcal{A}}(a)) = (\sigma \circ unfold^{\mathcal{A}})^\#(a) = g^\#(a) = t. \tag{17}$$

By (8), $unfold^{\mathcal{A}}(a) \in DTh(E) \cap DT_\Sigma$. Therefore, (17) implies $t \in CTh(E)$. ❑

19.18 Base algebra extensions

Let $\Sigma = (S, F)$ be a subsignature of a signature $\Sigma' = (S', F')$ and B be a Σ-algebra.

For all $e \in \mathcal{T}_{po}(S)$, $e_B \in \mathcal{T}_{po}(S)$ is obtained from e by replacing each sort $s \in S$ with B_s. Let $F_B = \{f_B : e_B \to e'_B \mid f : e \to e' \in F'\}$,

$$\Sigma_B = (S' \setminus S, F_B).$$

Moreover, $\sigma_B : \Sigma' \to \Sigma_B$ 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 Σ_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_Σ denote the forgetful functor from $Alg_{\Sigma'}$ to Alg_Σ, A be a Σ'-algebra and $B = U_\Sigma(A)$ (see section 5.1). A yields a Σ_B-algebra A_B that is defined as follows:

For all $s \in S' \setminus S$, $A_{B,s} = A_s$, and for all $f \in F'$, $f_B^{A_{B,s}} = f^{\mathcal{A}}$.

The σ_B-reduct of A_B agrees with A: $A_B|_{\sigma_B} = A$.

Let Σ_B be constructive and $\mu\Sigma$ be initial in Alg_{Σ_B}.

U_Σ has a left adjoint $L_{\Sigma'} : Alg_\Sigma \to Alg_{\Sigma'}$:

For all Σ-algebras B, $L_{\Sigma'}(B) =_{def} \mu\Sigma|_{\sigma_B}$ is called the **free Σ'-algebra over** B.

The unit $\eta : Id \to U_\Sigma L_{\Sigma'}$ is defined as follows: For all $b \in B$, $\eta_B(b) = b$.

The co-unit $\epsilon : L_{\Sigma'} U_\Sigma \to Id$ is defined as follows: For all Σ-algebras B and Σ'-algebras A,

$$L_{\Sigma'}(B) \xrightarrow{\epsilon_A} A \quad = \quad \mu\Sigma|_{\sigma_B} \xrightarrow{fold^{A_B}|_{\sigma_B}} A_B|_{\sigma_B}$$

where $fold^{A_B}$ is the unique Σ_B-homomorphism from $\mu\Sigma$ to A_B.

Let Σ_B be destructive and $\nu\Sigma$ be final in Alg_{Σ_B}.

U_Σ has a right adjoint $R_{\Sigma'} : Alg_\Sigma \to Alg_{\Sigma'}$:

For all Σ-algebras B, $R_{\Sigma'}(B) =_{def} \nu\Sigma|_{\sigma_B}$ is called the **cofree Σ'-algebra over** B.

The co-unit $\epsilon : U_\Sigma R_{\Sigma'} \to Id$ is defined as follows: For all $b \in B$, $\epsilon_B(b) = b$.

The unit $\eta : Id \to R_{\Sigma'} U_\Sigma$ is defined as follows: For all Σ-algebras B and Σ'-algebras A,

$$A \xrightarrow{\eta_A} R_{\Sigma'}(B) \quad = \quad A_B|_{\sigma_B} \xrightarrow{unfold^{A_B}|_{\sigma_B}} \nu\Sigma|_{\sigma_B}$$

where $unfold^{A_B}$ is the unique Σ_B-homomorphism from A_B to $\nu\Sigma$.

Let X be a semiring, $C\Sigma = (\{list\}, \{X\}, C, \emptyset)$, $\Sigma = (C\Sigma, Stream(X), \emptyset)$,

$$\begin{aligned} C \;=\; \{ & \underline{\_} : X \to list, \\ & \mathcal{X} : 1 \to list, \\ & \prec : X \times list \to list, \\ & +, *, \times, \otimes, \circ : list \times list \to list, \\ & \_^{-1} : list \to list, \\ & \_^{\underline{-1}} : list \to list, \\ & \textstyle\sum_{n<\omega} : list^{\mathbb{N}} \to list, \\ & exp, sin, cos : list \to list\} \end{aligned}$$

and E be the following system of recursive equations: Let $x, s, s' \in V$.

$$\begin{aligned} head(\underline{x}) &= x & tail(\underline{x}) &= \underline{0} \\ head(\mathcal{X}) &= 0 & tail(\mathcal{X}) &= \underline{1} \\ head(x \prec s) &= x & tail(x \prec s) &= s \\ head(s + s') &= head(s) + head(s') & tail(s + s') &= tail(s) + tail(s') \end{aligned}$$

$$head(s * s') = head(s) * head(s') \quad tail(s * s') = tail(s) * tail(s')$$

$$head(s \times s') = head(s) * head(s') \quad tail(s \times s') = (tail(s) \times s') + (\underline{head(s)} \times tail(s'))$$

convolution product

$$head(s \otimes s') = head(s) * head(s') \quad tail(s \otimes s') = (tail(s) \otimes s') + (s \otimes tail(s'))$$

shuffle product

$$head(s \circ s') = head(s) \qquad tail(s \circ s') = tail(s') \times (tail(s) \circ s')$$

$$head(s^{-1}) = head(s)^{-1} \qquad tail(s^{-1}) = (\underline{-1} \times \underline{head(s)^{-1}} \times tail(s)) \times s^{-1}$$

$$head(s^{\underline{-1}}) = head(s)^{-1} \qquad tail(s^{\underline{-1}}) = \underline{-1} \times (tail(s) \otimes s^{-1} \otimes s^{-1})$$

$$head(\textstyle\sum_{n<\omega} s_n) = \sum_{n<\omega} head(s_n) \quad tail(\sum_{n<\omega} s_n) = \sum_{n<\omega} tail(s_n)$$

$$head(exp(s)) = exp(head(s)) \qquad tail(exp(s)) = tail(s) \otimes exp(s)$$

$$head(sin(s)) = sin(head(s)) \qquad tail(sin(s)) = tail(s) \otimes cos(s)$$

$$head(cos(s)) = cos(head(s)) \qquad tail(cos(s)) = tail(s) \otimes -sin(s)$$

Let $a, b \in X^{\mathbb{N}}$. -1 and $head(a)^{-1}$ 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 [157], p. 14). $-a = \underline{-1} \times a$. $exp(a)$, $sin(a)$, $cos(a)$ are defined if $X \in \{\mathbb{R}, \mathbb{C}\}$.

Given $a_0, a_1, a_2, \dots \in X^{\mathbb{N}}$, $\sum_{n<\omega} head(a_n)$ is defined only if X is a complete semiring or $a_0, a_1, a_2, \dots$ is **summable**, i.e., for all $i \in \mathbb{N}$, $\sum_{n<\omega} a_n(i) = \lim_{n\to\infty} \sum_{k=0}^{n} a_k(i) \neq \infty$ (see [159], section 4).

For all $x \in X$, $a, b \in X^{\mathbb{N}}$ and $n \in \mathbb{N}$,

$$\underline{x}(n) = if\ n = 0\ then\ x\ else\ 0, \quad (1)$$

$$\mathcal{X}(n) = if\ n = 1\ then\ 1\ else\ 0, \quad (2)$$

$$(x \prec a)(n) = if\ n = 0\ then\ x\ else\ a(n-1), \quad (3)$$

$$(a + b)(n) = a(n) + b(n), \quad (4)$$

$$(a \times b)(n) = \sum_{i=0}^{n} a(i) * b(n-i) \quad (5)$$

$$(a \otimes b)(n) = \sum_{i=0}^{n} \binom{n}{i} * a(i) * b(n-i) \quad (6)$$

Proof.

Since $X^{\mathbb{N}}$ is final in $Alg_{Stream(X)}$, Theorem 16.3 implies that, if the interpretation of $\underline{\ }, \mathcal{X}, \prec, +, \times, \otimes$ 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:

$$head(\underline{x}) = \underline{x}(0) = x,$$

$$tail(\underline{x})(n) = \underline{x}(n+1) = 0 = \underline{0}(n),$$

$$head(\mathcal{X}) = \mathcal{X}(0) = 0,$$

$$tail(\mathcal{X})(n) = \mathcal{X}(n+1) = \mathit{if}\ n+1 = 1\ \mathit{then}\ 1\ \mathit{else}\ 0 = \mathit{if}\ n = 0\ \mathit{then}\ 1\ \mathit{else}\ 0 = \underline{1}(n),$$

$$head(x \prec a) = (x \prec a)(0) = x,$$

$$tail(x \prec a)(n) = (x \prec a)(n+1) = \mathit{if}\ n+1 = 0\ \mathit{then}\ x\ \mathit{else}\ a(n)$$

$$= \mathit{if}\ n = -1\ \mathit{then}\ x\ \mathit{else}\ a(n) = a(n)$$

$$head(a+b) = (a+b)(0) = a(0) + b(0) = head(a) + head(b),$$

$$tail(a+b)(n) = (a+b)(n+1) = a(n+1) + b(n+1) = tail(a)(n) + tail(b)(n)$$

$$= (tail(a) + tail(b))(n),$$

$$head(a \times b) = (a \times b)(0) = \sum_{i=0}^{0} a(i) * b(0-i) = a(0) * b(0) = head(a) * head(b),$$

$$tail(a \times b)(n) = (a \times b)(n+1) = \sum_{i=0}^{n+1} a(i) * b(n+1-i)$$

$$= a(0) * b(n+1) + \sum_{i=1}^{n+1} a(i) * b(n+1-i)$$

$$= a(0) * b(n+1) + \sum_{i=0}^{n} a(i+1) * b(n+1-(i+1))$$

$$= a(0) * b(n+1) + \sum_{i=0}^{n} a(i+1) * b(n-i)$$

$$= \sum_{i=0}^{n} a(i+1) * b(n-i) + a(0) * b(n+1)$$

$$= \sum_{i=0}^{n} a(i+1) * b(n-i) + a(0) * b(n+1) + \sum_{i=1}^{n} 0 * b(n-i+1)$$

$$= \sum_{i=0}^{n} a(i+1) * b(n-i) + \underline{a(0)}(0) * b(n+1) + \sum_{i=1}^{n} \underline{a(0)}(i) * b(n-i+1)$$

$$= \sum_{i=0}^{n} a(i+1) * b(n-i) + \sum_{i=0}^{n} \underline{a(0)}(i) * b(n-i+1)$$

$$= \sum_{i=0}^{n} tail(a)(i) * b(n-i) + \sum_{i=0}^{n} \underline{a(0)}(i) * tail(b)(n-i)$$

$$= (tail(a) \times b)(n) + (\underline{a(0)} \times tail(b))(n)$$

$$= ((tail(a) \times b) + (\underline{head(a)} \times tail(b)))(n),$$

$$head(a \otimes b) = (a \otimes b)(0) = \sum_{i=0}^{0} \binom{0}{i} * a(i) * b(0-i) = 1 * a(0) * b(0)$$

$$= a(0) * b(0) = head(a) * head(b),$$

$$tail(a \otimes b)(n) = (a \times b)(n+1) = \sum_{i=0}^{n+1} \binom{n+1}{i} * a(i) * b(n+1-i)$$
$$= \binom{n+1}{0} * a(0) * b(n+1) + \sum_{i=1}^{n+1} \binom{n+1}{i} * a(i) * b(n+1-i)$$
$$= a(0) * b(n+1) + \sum_{i=0}^{n} \binom{n+1}{i+1} * a(i+1) * b(n-i)$$
$$= a(0) * b(n+1) + \sum_{i=0}^{n} \binom{n}{i+1} * a(i+1) * b(n-i) + \sum_{i=0}^{n} \binom{n}{i} * a(i+1) * b(n-i)$$
$$= a(0) * b(n+1) + \sum_{i=0}^{n} \binom{n}{i+1} * a(i+1) * b(n-(i+1)+1)$$
$$+ \sum_{i=0}^{n} \binom{n}{i} * a(i+1) * b(n-i)$$
$$= \sum_{i=0}^{n} \binom{n}{i} * a(i) * b(n-i+1) + \sum_{i=0}^{n} \binom{n}{i} * a(i+1) * b(n-i)$$
$$= \sum_{i=0}^{n} \binom{n}{i} * a(i+1) * b(n-i) + \sum_{i=0}^{n} \binom{n}{i} * a(i) * b(n-i+1)$$
$$= \sum_{i=0}^{n} \binom{n}{i} * tail(a)(i) * b(n-i) + \sum_{i=0}^{n} \binom{n}{i} * a(i) * tail(b)(n-i)$$
$$= (tail(a) \otimes b)(n) + (a \otimes tail(b))(n) = ((tail(a) \otimes b) + (a \otimes tail(b)))(n). \qquad \square$$

For all $x \in X$, $a \in X^{\mathbb{N}}$ and summable $\{a_n\}_{n<\omega} \subseteq X^{\mathbb{N}}$,

$$
\begin{array}{lcll}
\underline{x} \times a & = & \lambda n.(x * a(n)) \;=\; \underline{x} \otimes a, & \text{([157], p. 24)} \\
tail(\underline{x} \times a) & = & \underline{x} \times tail(a), & \\
\mathcal{X} \times a & = & 0 \prec a, & \text{([141], equation (11))} \\
\underline{x} \times (y \times a) & = & \underline{x * y} \times a, & \\
\underline{x} \times (y \prec a) & = & x * y \prec (\underline{x} \times a), & \\
\sum_{n<\omega} a_n & = & a_0 + \sum_{n<\omega} a_{n+1}, & \text{(proof by coinduction)} \\
\sum_{n<\omega} a \times a_n & = & a \times \sum_{n<\omega} a_n. & \text{(proof by coinduction)}
\end{array}
$$

For all $n \in \mathbb{N}$, define $a^0 = a^{\underline{0}} = \underline{1}$, $a^{n+1} = a \times a^n$ and $a^{\underline{n+1}} = a \otimes a^{\underline{n}}$. By coinduction,

$$
a(0) = 0 \;\Rightarrow\; tail(a^{n+1}) = tail(a) \times a^n, \tag{7}
$$

$$
tail(a^{\underline{n+1}}) = \underline{n+1} \otimes tail(a) \otimes a^{\underline{n}}. \tag{8}
$$

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

$$\mathcal{X}^n \;=\; \lambda i.(\mathit{if}\ i = n\ \mathit{then}\ 1\ \mathit{else}\ 0), \qquad \mathcal{X}^{\underline{n}} \;=\; \underline{n!} \times \mathcal{X}^n$$

([157], Equation (30)) and thus for all $x \in X$:

$$\underline{x} \times \mathcal{X}^n \;=\; \lambda i.(\mathit{if}\ i = n\ \mathit{then}\ x\ \mathit{else}\ 0).$$

Representations of $X^{\mathbb{N}}$

Let X be a group. Then $B = X^{\mathbb{N}}$ is the carrier of a $\mathit{Stream}(X)$-algebra: $\mathit{head}^B(a) = a(0)$ and $\mathit{tail}^B(a) = \lambda n.(a(n+1) - a(n))$.

The function $\tau : \langle \mathit{head}^B, \mathit{tail}^B \rangle \to X^{\mathbb{N}}$ that maps $a \in X^{\mathbb{N}}$ to the stream $\lambda n. \sum_{i=0}^{n} \binom{-n}{i} * a(i)$ is a $\mathit{Stream}(X)$-isomorphism ([141], section 1.2).

The inverse of τ maps $a \in X^{\mathbb{N}}$ to $\lambda n. \sum_{i=0}^{n} \binom{n}{i} * a(i)$.

The set $\mathbb{A}$ of functions $f : \mathbb{R} \to \mathbb{R}$ that are analytic at 0 provides the carrier of a $\mathit{Stream}(X)$-algebra: $\mathit{head}^{\mathbb{A}}(f) = f(0)$ and $\mathit{tail}^{\mathbb{A}}(f) = Df$ (first derivative of f).

The **Taylor transform** $T : \mathbb{A} \to \mathbb{R}^{\mathbb{N}}$ that maps $f \in \mathbb{A}$ to the stream $\lambda n.(D^n f)(0)$ is a $\mathit{Stream}(X)$-monomorphism ([141], section 2.2; [157], section 5; [158], section 3.4; [159], 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.

$$\begin{aligned} \mathit{exp} &= \lambda x. \textstyle\sum_{n<\omega} a(n) * x^n \text{ where } a(n) = 1/n! \\ \mathit{sin} &= \lambda x. \textstyle\sum_{n<\omega} a(n) * x^n \text{ where } a(n) = \mathit{if}\ \mathit{even}(n)\ \mathit{then}\ 0\ \mathit{else}\ (-1)^{n//2}/n!) \\ \mathit{cos} &= \lambda x. \textstyle\sum_{n<\omega} a(n) * x^n \text{ where } a(n) = \mathit{if}\ \mathit{odd}(n)\ \mathit{then}\ 0\ \mathit{else}\ (-1)^{n//2}/n!) \end{aligned}$$

Proofs by coinduction yield for all $f, g \in \mathbb{A}$:

$$T(f+g) = T(f) + T(g) \tag{9}$$
$$T(f * g) = T(f) \otimes T(g) \tag{10}$$
$$T(\lambda x. \int_0^x f) = \mathcal{X} \times T(f) \tag{11}$$

The function $g : \mathbb{R}^{\mathbb{N}} \to \mathbb{R}^{\mathbb{N}}$ 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 $\underline{\underline{\ }}$, $\mathcal{X}$, $\prec$ and $+$ and satisfies

$$g(a \times b) = g(a) \otimes g(b) \quad \text{and} \quad g(a^{-1}) = g(a)^{\underline{-1}}$$

([141], section 3.1; [157], Thm. 6.4 where $g = \Delta_{\mathbb{R}^{\mathbb{N}}}$; [159], 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** $\lambda x. \sum_{n<\omega} a(n) * x^n$ ([141], section 3.1). The inverse $g^{-1} \circ T$ sends $f \in \mathbb{A}$ to $\lambda n.(D^n f/n!)$.

Theorem 20.1 (fundamental theorem of *Stream*(X))

Let A be the final *Stream*(X)-algebra (with carrier $X^{\mathbb{N}}$). A satisfies the following set E of equations: Let $s, s' \in V$ and for all $n \in \mathbb{N}$, $s(n) = head(tail^n(s))$.

$$
\begin{aligned}
s &= \underline{head(s)} + (\mathcal{X} \times tail(s)), && (12)\\
s &= \sum_{n<\omega} \underline{s(n)} \times \mathcal{X}^n, && (13)\\
s &= \sum_{n<\omega} \underline{s(n)/n!} \times \mathcal{X}^{\underline{n}}, && (14)\\
head(s') = 0 \quad \Rightarrow \quad s \circ s' &= \sum_{n<\omega} \underline{s(n)} \times s'^n, && (15)\\
exp(s) &= \sum_{n<\omega} \underline{1/n!} \times s^{\underline{n}}, && (16)\\
exp(s+s') &= exp(s) \otimes exp(s'), && (17)\\
sin(s) &= \sum_{n<\omega} \underline{(-1)^n/(2n+1)!} \times s^{\underline{2n+1}}, && (18)\\
cos(s) &= \sum_{n<\omega} \underline{(-1)^n/(2n)!} \times s^{\underline{2n}}, && (19)
\end{aligned}
$$

$$sin(s)^{\underline{2}} + cos(s)^{\underline{2}} \;=\; \underline{1}. \tag{20}$$

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 [158], 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 Σ with their interpretations in A.

(12)

$$head(\underline{head(a)}+(\mathcal{X}\times tail(a)))=head(\underline{a(0)})+head(\mathcal{X}\times tail(a))$$

$$=head(\underline{a(0)})+head(0\prec tail(a))=head(\underline{a(0)})+0=head(\underline{a(0)})=a(0)=head(a),$$

$$tail(\underline{head(a)}+(\mathcal{X}\times tail(a)))=tail(\underline{a(0)})+tail(\mathcal{X}\times tail(a))=\underline{0}+tail(\mathcal{X}\times tail(a))$$

$$=tail(\mathcal{X}\times tail(a))=tail(0\prec tail(a))=tail(a)$$

(see [157], Thm. 5.1; [159], Thm. 4.1).

(13)

$$head(\sum_{n<\omega} \underline{a(n)} \times \mathcal{X}^n) = \sum_{n<\omega} head(\underline{a(n)} \times \mathcal{X}^n) = \sum_{n<\omega} head(\underline{a(n)}) * head(\mathcal{X}^n)$$

$$= \sum_{n<\omega} a(n) * head(\mathcal{X}^n) = \sum_{n<\omega} a(n) * \mathcal{X}^n(0) = a(0) = head(a),$$

$$tail(\sum_{n<\omega} \underline{a(n)} \times \mathcal{X}^n) = \sum_{n<\omega} tail(\underline{a(n)} \times \mathcal{X}^n) = \sum_{n<\omega} \underline{a(n)} \times tail(\mathcal{X}^n)$$

$$= (\underline{a(0)} \times tail(\mathcal{X}^0)) + \sum_{n<\omega} \underline{a(n+1)} \times tail(\mathcal{X}^{n+1})$$

$$= (\underline{a(0)} \times \underline{0}) + \sum_{n<\omega} \underline{a(n+1)} \times \mathcal{X}^n = \sum_{n<\omega} \underline{a(n+1)} \times \mathcal{X}^n$$

$$= \sum_{n<\omega} \underline{tail(a)(n)} \times \mathcal{X}^n \sim tail(a)$$

(see [157], Thm. 5.2; [159], Thm. 4.3).

(14) By (13),

$$a = \sum_{n<\omega} \underline{a(n)} \times \mathcal{X}^n = \sum_{n<\omega} \underline{n! * a(n)/n!} \times \mathcal{X}^n = \sum_{n<\omega} \underline{a(n)/n!} \times (\underline{n!} \times \mathcal{X}^n)$$

$$= \sum_{n<\omega} \underline{a(n)/n!} \times \mathcal{X}^{\underline{n}}.$$

(15)

$$head(\sum_{n<\omega} \underline{a(n)} \times b^n) = \sum_{n<\omega} head(\underline{a(n)} \times b^n) = \sum_{n<\omega} head(\underline{a(n)}) * head(b^n)$$
$$= \sum_{n<\omega} a(n) * b^n(0) = \sum_{n<\omega} a(n) * b(0)^n \overset{b(0)=0}{=} a(0) * 1 = a(0) = head(a) = head(a \circ b),$$

$$tail(\sum_{n<\omega} \underline{a(n)} \times b^n) = \sum_{n<\omega} tail(\underline{a(n)} \times b^n) = \sum_{n<\omega} \underline{a(n)} \times tail(b^n)$$
$$= (\underline{a(0)} \times tail(b^0)) + \sum_{n<\omega} \underline{a(n+1)} \times tail(b^{n+1})$$
$$= (\underline{a(0)} \times \underline{0}) + \sum_{n<\omega} \underline{a(n+1)} \times tail(b^{n+1}) = \sum_{n<\omega} \underline{a(n+1)} \times tail(b^{n+1})$$
$$\overset{(7)}{=} \sum_{n<\omega} \underline{tail(a)(n)} \times (tail(b) \times b^n) = tail(b) \times \sum_{n<\omega} \underline{tail(a)(n)} \times b^n$$
$$\sim_C tail(b) \times (tail(a) \circ b) = tail(a \circ b)$$

(see [158], Thm. 2.5.3).

(16)

$$head(\sum_{n<\omega} \underline{1/n!} \times a^{\underline{n}}) = \sum_{n<\omega} head(\underline{1/n!} \times a^{\underline{n}}) = \sum_{n<\omega} head(\underline{1/n!}) * head(a^{\underline{n}})$$

$$= \sum_{n<\omega}(1/n!) * a^{\underline{n}}(0) = \sum_{n<\omega}(1/n!) * a(0)^n = exp(a(0)) = head(exp(a)),$$

$$tail(\sum_{n<\omega} \underline{1/n!} \times a^{\underline{n}}) = \sum_{n<\omega} tail(\underline{1/n!} \times a^{\underline{n}}) = \sum_{n<\omega} \underline{1/n!} \times tail(a^{\underline{n}})$$

$$= \underline{1/0!} \times tail(a^{\underline{0}}) + \sum_{n<\omega} \underline{1/(n+1)!} \times tail(a^{\underline{n+1}})$$

$$= \underline{1} \times \underline{0} + \sum_{n<\omega} \underline{1/(n+1)!} \times tail(a^{\underline{n+1}}) = \sum_{n<\omega} \underline{1/(n+1)!} \times tail(a^{\underline{n+1}})$$

$$\stackrel{(8)}{=} \sum_{n<\omega} \underline{1/(n+1)!} \times (\underline{n+1} \otimes tail(a) \otimes a^{\underline{n}})$$

$$= \sum_{n<\omega} \underline{1/(n+1)!} \otimes \underline{n+1} \otimes tail(a) \otimes a^{\underline{n}}$$

$$= tail(a) \otimes \sum_{n<\omega} \underline{1/(n+1)!} \otimes \underline{n+1} \otimes a^{\underline{n}}$$

$$= tail(a) \otimes \sum_{n<\omega} \underline{(n+1)/(n+1)!} \otimes a^{\underline{n}} = tail(a) \otimes \sum_{n<\omega} \underline{1/n!} \otimes a^{\underline{n}}$$

$$\sim_C tail(a) \otimes exp(a) = tail(exp(a)).$$

(19)

$$head(exp(a+b)) = exp(head(a+b)) = exp(head(a)+head(b))$$

$$= exp(head(a)) * exp(head(b)) = head(exp(a)) * head(exp(b))$$

$$= head(exp(a) \otimes exp(b)),$$

$$tail(exp(a+b)) = tail(a+b) \otimes exp(a+b) = (tail(a)+tail(b)) \otimes exp(a+b)$$

$$= (tail(a) \otimes exp(a+b)) + (tail(b) \otimes exp(a+b))$$

$$\sim_C (tail(a) \otimes exp(a) \otimes exp(b)) + (tail(b) \otimes exp(a) \otimes exp(b))$$

$$= (tail(a) \otimes exp(a) \otimes exp(b)) + (exp(a) \otimes tail(b) \otimes exp(b))$$

$$= (tail(exp(a)) \otimes exp(b)) + (exp(a) \otimes tail(exp(b)))$$

$$= tail(exp(a) \otimes exp(b)).$$

(20)

$$head(sin(a)^{\underline{2}} + cos(a)^{\underline{2}}) = head(sin(a)^{\underline{2}}) + head(cos(a)^{\underline{2}})$$

$$= head(sin(a) \otimes sin(a)) + head(cos(a) \otimes cos(a))$$

$$= head(sin(a))^2 + head(cos(a))^2 = sin(head(a))^2 + cos(head(a))^2 = 1 = head(\underline{1}),$$

$$
\begin{aligned}
& tail(sin(a)^{\underline{2}} + cos(a)^{\underline{2}}) = tail(sin(a)^{\underline{2}}) + tail(cos(a)^{\underline{2}}) \\
& = tail(sin(a) \otimes sin(a)) + tail(cos(a) \otimes cos(a)) \\
& = (tail(sin(a)) \otimes sin(a)) + (sin(a) \otimes tail(sin(a))) \\
& \quad + (tail(cos(a)) \otimes cos(a)) + (cos(a) \otimes tail(cos(a))) \\
& = (tail(a) \otimes cos(a) \otimes sin(a)) + (sin(a) \otimes tail(a) \otimes cos(a)) \\
& \quad + (tail(a) \otimes -sin(a) \otimes cos(a)) + (cos(a) \otimes tail(a) \otimes -sin(a)) \\
& = (tail(a) \otimes cos(a) \otimes sin(a)) - (tail(a) \otimes sin(a) \otimes cos(a)) \\
& \quad + (sin(a) \otimes tail(a) \otimes cos(a)) - (cos(a) \otimes tail(a) \otimes sin(a)) \\
& = (sin(a) \otimes (tail(sin(a)) + tail(sin(a)))) + (cos(a) \otimes (tail(cos(a)) + tail(cos(a)))) \\
& = (sin(a) \otimes ((tail(a) \otimes cos(a)) + (tail(a) \otimes cos(a)))) \\
& \quad + (cos(a) \otimes ((tail(a) \otimes -sin(a)) + (tail(a) \otimes -sin(a)))) \\
& = \underline{0} + \underline{0} = \underline{0} = tail(\underline{1})
\end{aligned}
$$

(see [48], p. 32). ❑

Theorem 20.2 (fundamental theorem of calculus)

For all $f \in \mathbb{A}$,

$$f \;=\; \lambda x.(f(0) + \int_0^x Df). \tag{21}$$

Proof.

$$T(\lambda x.(f(0) + \int_0^x Df)) = T(\lambda x.f(0) + \lambda x.\int_0^x Df) \overset{(9)}{=} T(\lambda x.f(0)) + T(\lambda x.\int_0^x Df)$$

$$\overset{(11)}{=} \underline{f(0)} + (\mathcal{X} \times T(Df)) = \underline{f(0)} + (0 \prec T(Df)) = f(0) \prec T(Df)$$

$$= head(T(f)) \prec tail(T(f)) = T(f).$$

Since T is mono, (21) holds true. ❏

Example Fibonacci sequence

$$fib(0) = 0 \;\wedge\; fib(1) = 1 \;\wedge\; fib = tail(tail(fib)) - tail(fib)$$

$$\iff \quad fib = 0 \prec fib + (1 \prec fib)$$

$$\iff \quad fib = 0 \prec fib' \;\wedge\; fib' = 1 \prec fib + fib' \qquad \text{([73], p. 9)}$$

$$\iff \quad fib = \mathcal{X} \times (\underline{1} - \mathcal{X} - \mathcal{X}^2)^{-1} \qquad \text{([159], p. 111)}$$

$$\iff \quad fib = 0 \prec ((\underline{1} + \mathcal{X}) \times fib) + \underline{1} \qquad \text{([48], p. 33)}$$

21 Conservative extensions

Let $\Sigma = (S, F, P)$ be a signature, $\Sigma' = (S', F', P')$ be a subsignature of Σ, AX be a set of Σ-formulas, $AX' \subseteq AX$ be a set Σ'-formulas, A be a Σ-algebra and $B = A|_{\Sigma'}$.

21.1 Constructor extensions

Let Σ 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 $fold^B : \mu\Sigma' \to B$ is surjective.
A is **equationally F'-consistent** if $fold^B$ is injective.

(Σ, AX) is a **conservative extension of** (Σ', 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 Σ-algebras, the full subcategory of F-reachable objects of $\mathcal{K}$ is denoted by $gen(\mathcal{K})$.

Lemma 21.1

Let A be initial in $Alg_{\Sigma,AX}$.

A is F'-reachable iff $img(fold^B)$ is a Σ-invariant.

Proof. "$\Rightarrow$": Let A be F'-reachable. Then $img(fold^B) = B = A$ and thus $img(fold^B)$ is a Σ-invariant.

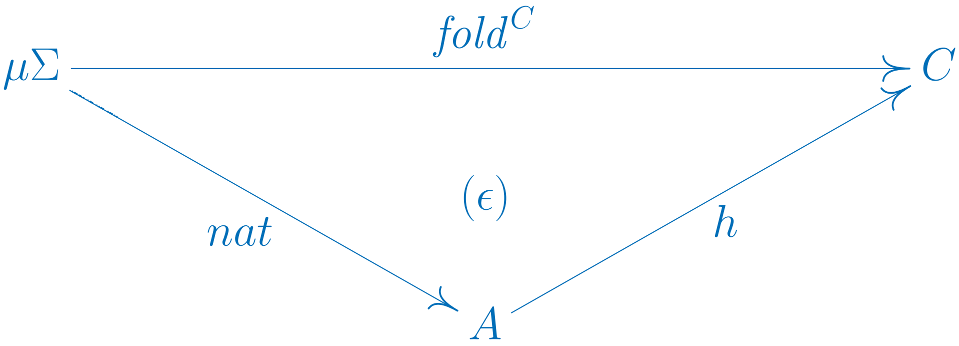
"$\Leftarrow$": Let $img(fold^B)$ be a Σ-invariant. By Lemma 12.3 (1), A is the least Σ-invariant of A. Hence $B = A \subseteq img(fold^B) \subseteq B$ and thus by (1), A is F'-reachable. ❑

Lemma 21.2

Let $\mu\Sigma'$ be extendable to a (Σ, AX)-algebra C.

Then (Σ, AX) is a conservative extension of (Σ', AX').

Proof. Let $fold^C$ be the unique Σ-homomorphism from $\mu\Sigma$ to C, $A = \mu\Sigma/ker(fold^C)$ and $B = A|_{\Sigma'}$. By Lemma 13.1 (2), there is a unique Σ-monomorphism $h : A \to C$ such that (ϵ) commutes:



By Lemma 13.6 (4), A satisfies AX. Hence $A \in Alg_{\Sigma,AX}$ and thus $B \in Alg_{\Sigma',AX'}$. Let $fold^B$ be the unique Σ'-homomorphism from $\mu\Sigma'$ to B.

$$\mu\Sigma' \stackrel{fold^B}{\to} B \stackrel{h|_{\Sigma'}}{\to} C|_{\Sigma'} = \mu\Sigma'$$

agrees with the identity on $\mu\Sigma'$ because $\mu\Sigma'$ is initial. Since $id_{\mu\Sigma'}$ is epi, Lemma 4.1 (1) implies that $h|_{\Sigma'}$ is also epi. We conclude that $\mu\Sigma'$ and B are Σ'-isomorphic and thus (Σ, AX) is a conservative extension of (Σ', AX'). ❏

21.2 Destructor extensions

Let Σ 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 $unfold^B : B \to \nu\Sigma'$ is injective.
A is **behaviorally F'-complete** if $unfold^B$ is surjective.

(Σ, AX) is a **conservative extension** of (Σ', 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 Σ-algebras, the full subcategory of F-observable objects of $\mathcal{K}$ is denoted by $obs(\mathcal{K})$.

Lemma 21.3

Let A be final in $Alg_{\Sigma,AX}$.

A is F'-observable iff $ker(unfold^B)$ is a Σ-congruence.

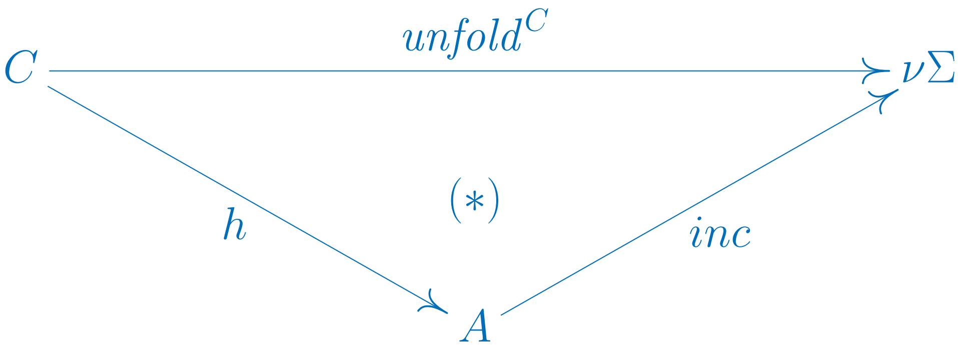
Proof. "$\Rightarrow$": Let A be F'-observable. Then $ker(unfold^B) = \Delta_B = \Delta_A$ and thus $ker(unfold^B)$ is a Σ-congruence.

"$\Leftarrow$": Let $ker(unfold^B)$ be a Σ-congruence. By Lemma 13.3 (1), Δ_A is the greatest Σ-congruence on A. Hence $\Delta_B \subseteq ker(unfold^B) \subseteq \Delta_A = \Delta_B$ and thus by (3), A is F'-observable. ❑

Lemma 21.4

Let $\nu\Sigma'$ be extendable to a (Σ, AX)-algebra C. Then (Σ, AX) is a conservative extension of (Σ', AX').

Proof. Let $unfold^C$ be the unique Σ-homomorphism from C to $\nu\Sigma$, $A = img(unfold^C)$ and $B = A|_{\Sigma'}$. By Lemma 12.1 (2), there is a unique Σ-epimorphism $h : C \to A$ such that $(*)$ commutes:



By Lemma 12.7 (2), A satisfies AX. Hence $A \in Alg_{\Sigma,AX}$ and thus $B \in Alg_{\Sigma',AX'}$. Let $unfold^B$ be the unique Σ'-homomorphism from B to $\mu\Sigma'$.

$$\nu\Sigma' = C|_{\Sigma'} \overset{h|_{\Sigma'}}{\to} B \overset{unfold^B}{\to} \nu\Sigma'$$

agrees with the identity on $\nu\Sigma'$ because $\nu\Sigma'$ is final. Since $id_{\nu\Sigma'}$ is mono, Lemma 4.1 (2) implies that $h|_{\Sigma'}$ is also mono. We conclude that $\nu\Sigma'$ and B are Σ'-isomorphic and thus (Σ, AX) is a conservative extension of (Σ', AX'). ❑

Let $\Sigma = (S, F, P)$ be a constructor signature, ??? $\Sigma' = (S, F)$ and $\mu\Sigma'$ be initial in $Alg_{\Sigma'}$.

Lemma 22.1

Let $h : A \to B$ be a Σ-homomorphism that preserves all $p : e \in P$, i.e.,

$$p^A \;=\; \{a \in A_e \mid h(a) \in p^B\},$$

$e = \prod_{x \in V} e_x \in \mathcal{T}_{po}(S)$ and φ be a negation-free Σ-formula over V.

If φ 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 φ.

Let $p : e \in P$, $x \in V_s$. W.l.o.g. we assume that r is unary.

$$f \in r(t)^A \;\Leftrightarrow\; t^A(f) \in r^A \;\Leftrightarrow\; t^B(h \circ f) \overset{Lemma\ 9.9}{=} h(t^A(f)) \in r^B \;\Leftrightarrow\; h \circ f \in r(t)^B.$$

$$f \in (\varphi \wedge \psi)^A = \varphi^{\mathcal{A}} \cap \psi^A \overset{i.h.}{\Rightarrow} h \circ f \in \varphi^{\mathcal{B}} \cap \psi^B = (\varphi \wedge \psi)^B.$$

$$f \in (\varphi \vee \psi)^A = \varphi^{\mathcal{A}} \cup \psi^A \overset{i.h.}{\Rightarrow} h \circ f \in \varphi^{\mathcal{B}} \cup \psi^B = (\varphi \vee \psi)^B.$$

$$\begin{aligned} f \in (\exists x \varphi)^A \;&\Leftrightarrow\; \exists\, a \in A_s : upd(f,x,a) \in \varphi^{\mathcal{A}} \\ &\overset{i.h.}{\Rightarrow}\; \exists\, a \in A_s : upd(h \circ f, x, h(a)) = h \circ upd(f,x,a) \in \varphi^{\mathcal{B}} \\ &\Rightarrow\; \exists\, b \in B_s : upd(h \circ f, x, b) \in \varphi^{\mathcal{B}} \;\Leftrightarrow\; h \circ f \in (\exists x \varphi)^B. \end{aligned}$$

Proof of (2) by induction on the size of φ.

Let $r \in R$, $s \in S$ and $x \in V_s$. W.l.o.g. we assume that r is unary.

$$h \circ f \in r(t)^B \;\Leftrightarrow\; h(t^A(f)) \overset{\text{Lemma 9.9}}{=} t^B(h \circ f) \in r^B \;\Leftrightarrow\; t^A(f) \in r^A \;\Leftrightarrow\; f \in r(t)^A.$$

$$h \circ f \in (\varphi \wedge \psi)^B = \varphi^{\mathcal{B}} \cap \psi^B \overset{i.h.}{\Rightarrow} f \in \varphi^{\mathcal{A}} \cap \psi^A = (\varphi \wedge \psi)^A.$$

$$h \circ f \in (\varphi \vee \psi)^B = \varphi^{\mathcal{B}} \cup \psi^B \overset{i.h.}{\Rightarrow} f \in \varphi^{\mathcal{A}} \cup \psi^A = (\varphi \vee \psi)^A.$$

$$h \circ f \in (\exists x\varphi)^B \;\Leftrightarrow\; \exists\, b \in B_s : upd(h \circ f, x, b) \in \varphi^{\mathcal{B}}$$

$$\overset{h\ epi}{\Rightarrow} \exists\, a \in A_s : h \circ upd(f, x, a) = upd(h \circ f, x, h(a)) \in \varphi^{\mathcal{B}}$$

$$\overset{i.h.}{\Rightarrow} \exists\, a \in A_s : upd(f, x, a) \in \varphi^{\mathcal{A}} \;\Leftrightarrow\; f \in (\exists x\varphi)^A.$$

$$h \circ f \in (\forall x\varphi)^B \;\Leftrightarrow\; \forall\, b \in B_s : upd(h \circ f, x, b) \in \varphi^{\mathcal{B}}$$

$$\Rightarrow \forall\, a \in A_s : h \circ upd(f, x, a) = upd(h \circ f, x, h(a)) \in \varphi^{\mathcal{B}}$$

$$\overset{i.h.}{\Rightarrow} \forall\, a \in A_s : upd(f, x, a) \in \varphi^{\mathcal{A}} \;\Leftrightarrow\; f \in (\forall x\varphi)^A.$$

❑

22.1 Abstraction with a least congruence

Let AX consist of ∀-free Horn clauses and $\mathcal{K} = Alg_{\Sigma,AX}$ such that for all $A \in \mathcal{K}$, $=^A$ is a Σ-congruence, and $C = lfp(\mu\Sigma', \Sigma, AX)$.

Then $\sim \ =_{def} \ =^C$ is the least Σ-congruence on $\mu\Sigma'$.

By Lemma 13.6, $C/{\sim} \in \mathcal{K}$.

Let $A \in \mathcal{K}$. We define $B \in Alg_\Sigma$ as the $fold^A$-pre-image of the interpretation of R in A, i.e., for all $r : w \in R$,

$$r^B \ =_{def} \ \{b \in \mu\Sigma'_w \mid fold^A(b) \in r^A\}.$$

Use induction on $\mathbb{N}$ and Theorem 3.4 (or transfinite induction and Theorem 3.8) to show that $fold^A$ extends to a Σ-homomorphism!

B satisfies AX and thus $B \in Alg_{\Sigma,AX}$.

Proof. Let $\varphi = (r(t_1,\dots,t_n) \Leftarrow \psi) \in AX$ and $g \in \psi^B$. By Lemma 22.1 (1), $fold^A \circ g \in \psi^A$. Since A satisfies φ, $fold^A \circ g \in r(t_1,\dots,t_n)^A$, i.e.,

$$(fold^A(t_1^B(g)),\dots,fold^A(t_n^B(g))) \overset{Lemma\ 9.9}{=} (t_1^A(fold^A \circ g),\dots,t_n^A(fold^A \circ g)) \in r^A.$$

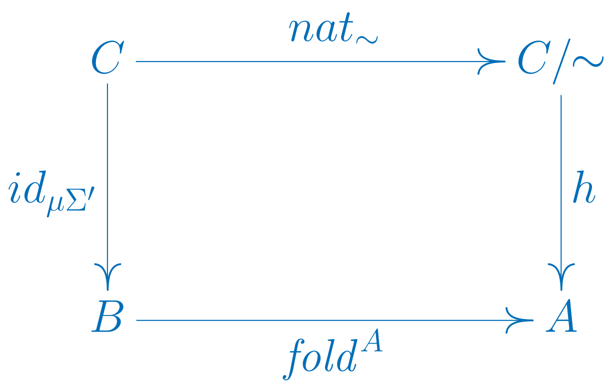
Hence $(t_1^B(g),\dots,t_n^B(g)) \in r^B$ and thus $g \in r(t_1,\dots,t_n)^B$. ❑

Theorem 22.2 $C/\sim$ is initial in $\mathcal{K}$.

Proof. Since C is the least $D \in Alg_{\Sigma,AX}$ with $D|_{\Sigma'} = \mu\Sigma'$, we obtain $C \leq B$. In particular,

$$\begin{aligned} \sim \;=\; =^C \;\subseteq\; =^B \;&=\; \{(t,u) \in (\mu\Sigma')^2 \mid fold^A(t) =^A fold^A(u)\} \\ &=\; ker(fold^A) \end{aligned}$$

because $=^A\, = \Delta_A$. Hence $h : C/\sim \to A$ is well-defined by $h \circ nat_\sim = fold^A \circ id_{\mu\Sigma'}$.



Since $nat_\sim$ is epi and reflects predicates and $fold^A \circ id_{\mu\Sigma'}$ is Σ-homomorphic, Lemma 9.1 (1) implies that h is also Σ-homomorphic.

Let h' be any Σ-homomorphism from $C/\sim$ to A. Since $B|_{B\Sigma} = BA$ is initial in Alg_Σ, $h' \circ nat_\sim = h \circ nat_\sim$ and thus $h' = h$ because $nat_\sim$ is epi. ❏

22.2 Abstraction with a greatest congruence

Let AX consist of co-Horn clauses and $\mathcal{K} = Alg_{\Sigma,AX}$ such that for all $A \in \mathcal{K}$, $=^A$ is a Σ-congruence, $C = gfp(\mu\Sigma', \Sigma, AX)$ and $\sim\ =_{def}\ =^C$ be a Σ-congruence on $\mu\Sigma'$. Hence $C \in gen(\mathcal{K})$.

By Lemma 13.6, $C/\!\sim\ \in gen(\mathcal{K})$.

Let $A \in gen(\mathcal{K})$. We define $B \in Alg_\Sigma$ as the $fold^A$-pre-image of the interpretation of R in A, i.e., for all $r : w \in R$,

$$r^B =_{def} \{b \in \mu\Sigma'_w \mid fold^A(b) \in r^A\}.$$

Use induction on $\mathbb{N}$ and Theorem 3.4 (or transfinite induction and Theorem 3.8) to show that $fold^A$ extends to a Σ-homomorphism!

B satisfies AX and thus $B \in gen(\mathcal{K})$.

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

$$(t_1^A(fold^A \circ g),\dots,t_n^A(fold^A \circ g)) \stackrel{Lemma\ 9.9}{=} (fold^A(t_1^B(g)),\dots,fold^A(t_n^B(g))) \in r^A.$$

Hence $fold^A \circ g \in r(t_1,\dots,t_n)^A$. Since A satisfies φ, $fold^A \circ g \in \psi^A$. Since A is Σ-reachable, $fold^A$ is epi and thus Lemma 22.1 (2) implies $g \in \psi^B$. ❑

Theorem 22.3 $C/{\sim}$ is final in $gen(\mathcal{K})$.

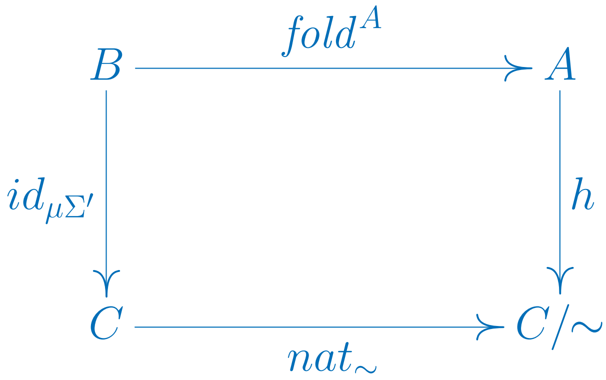
Proof. Since C is the greatest $D \in Alg_{\Sigma,AX}$ with $D|_{B\Sigma} = \mu\Sigma'$, we obtain $B \le C$. In particular,

$$ker(fold^A) = \{(t,u) \in (\mu\Sigma')^2 \mid fold^A(t) =^A fold^A(u)\} = =^B \subseteq =^C = \sim$$

because $=^A = \Delta_A$.

Hence for all $t,u \in \mu\Sigma'$, $fold^A(t) = fold^A(u)$ implies $t \sim u$. Since A is Σ-reachable, $fold^A$ is epi and thus for all $a \in A$ there is $t \in \mu\Sigma'$ with $fold^A(t) = a$.

Hence $h : A \to C/{\sim}$ is well-defined by $h \circ fold^A = nat_\sim \circ id_{\mu\Sigma'}$.



Since $fold^A$ is epi and reflects predicates and $nat_\sim \circ id_{\mu\Sigma'}$ is Σ-homomorphic, Lemma 9.1 (1) implies that h is also $\Sigma_{BA'}$-homomorphic.

Let h' be any Σ-homomorphism from A to $C/{\sim}$. Since $B|_{B\Sigma} = \nu\Sigma'$ is initial in Alg_Σ, $h' \circ fold^A = h \circ fold^A$ and thus $h' = h$ because $fold^A$ is epi. ❑

Let $\Sigma = (S, F, P)$ be a destructor signature, ??? $\Sigma' = (S, F)$ and $\nu\Sigma'$ be final in $Alg_{\Sigma'}$.

Lemma 22.4

Let $h : A \to B$ be a Σ-homomorphism that preserves all $p \in P$, i.e.,

$$p^B \;=\; h(p^A)$$

and φ be a negation-free Σ-formula.

If φ does not contain universal quantifiers, then

$$f \in \varphi^{\mathcal{A}} \quad \text{implies} \quad h \circ f \in \varphi^{\mathcal{B}}. \tag{3}$$

If h is mono and for all atomic subformulas $r(t_1, \ldots, t_n)$ of φ, $t_1, \ldots, t_n$ are variables, then

$$g \in \varphi^{\mathcal{B}} \quad \text{implies} \quad \exists\, f \in \varphi^{\mathcal{A}} : h \circ f =_{free(\varphi)} g. \tag{4}$$

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 φ.

$$f \in r(t)^A \;\Leftrightarrow\; t^A(f) \in r^A \;\Leftrightarrow\; t^B(h \circ f) \overset{Lemma\ 9.9}{=} h(t^A(f)) \in r^B \;\Leftrightarrow\; h \circ f \in r(t)^B.$$

$$f \in (\varphi \wedge \psi)^A = \varphi^{\mathcal{A}} \cap \psi^A \overset{i.h.}{\Rightarrow} h \circ f \in \varphi^{\mathcal{B}} \cap \psi^B = (\varphi \wedge \psi)^B.$$

$$f \in (\varphi \vee \psi)^A = \varphi^{\mathcal{A}} \cup \psi^A \overset{i.h.}{\Rightarrow} h \circ f \in \varphi^{\mathcal{B}} \cup \psi^B = (\varphi \vee \psi)^B.$$

$$\begin{aligned} f \in (\exists x\varphi)^A &\Leftrightarrow \exists\, a \in A_s : upd(f, x, a) \in \varphi^{\mathcal{A}} \\ &\overset{i.h.}{\Rightarrow} \exists\, a \in A_s : upd(h \circ f, x, h(a)) = h \circ upd(f, x, a) \in \varphi^{\mathcal{B}} \\ &\Rightarrow \exists\, b \in B_s : upd(h \circ f, x, b) \in \varphi^{\mathcal{B}} \Leftrightarrow h \circ f \in (\exists x\varphi)^B. \end{aligned}$$

Proof of (4) by induction on the size of φ*.*

Let $r \in R$, $s \in S$ and $x \in V_s$. W.l.o.g. we assume that r is unary.

$$\begin{aligned} g \in r(z)^B &\Leftrightarrow g(z) \in r^B \Leftrightarrow \exists\, a \in r^A : h(a) = g(z) \\ &\Leftrightarrow \exists\, f \in A^X : f(z) \in r^A \wedge h \circ f =_{\{z\}} g \Leftrightarrow \exists\, f \in r(z)^A : h \circ f =_{free(r(z))} g. \end{aligned}$$

$$\begin{aligned} g \in (\varphi \wedge \psi)^B = \varphi^{\mathcal{B}} \cap \psi^B &\overset{i.h.}{\Rightarrow} \exists\, f \in \varphi^{\mathcal{A}} : h \circ f =_{free(\varphi)} g \wedge \exists\, f' \in \psi^A : h \circ f' =_{free(\psi)} g \\ &\overset{h\ mono}{\Rightarrow} \exists\, f \in \varphi^{\mathcal{A}} \cap \psi^A : h \circ f =_{free(\varphi) \cup free(\psi)} g \\ &\Leftrightarrow \exists\, f \in (\varphi \wedge \psi)^A : h \circ f =_{free(\varphi \wedge \psi)} g. \end{aligned}$$

$$g \in (\varphi \vee \psi)^B \overset{analogously}{\Rightarrow} \exists\, f \in (\varphi \vee \psi)^A : h \circ f = g.$$

$$\begin{aligned}
g \in (\exists x \varphi)^B \;&\Leftrightarrow\; \exists\, b \in B_s : upd(g,x,b) \in \varphi^{\mathcal{B}} \\
&\overset{i.h.}{\Rightarrow}\; \exists\, b \in B_s : \exists\, f \in \varphi^{\mathcal{A}} : h \circ f =_{\mathit{free}(\varphi)} upd(g,x,b) \\
&\Rightarrow\; \exists\, f \in A^X : \exists\, a \in A_s : upd(f,x,a) \in \varphi^{\mathcal{A}} \wedge h \circ f =_{\mathit{free}(\varphi)\setminus\{x\}} g \\
&\Rightarrow\; \exists\, f \in (\exists x \varphi)^A : h \circ f =_{\mathit{free}(\exists x \varphi)} g.
\end{aligned}$$

$$\begin{aligned}
g \in (\forall x \varphi)^B \;&\Leftrightarrow\; \forall\, b \in B_s : upd(g,x,b) \in \varphi^{\mathcal{B}} \\
&\overset{i.h.}{\Rightarrow}\; \forall\, b \in B_s : \exists\, f \in \varphi^{\mathcal{A}} : h \circ f =_{\mathit{free}(\varphi)} upd(g,x,b) \\
&\overset{h\ mono}{\Rightarrow}\; \exists\, f \in A^X : \forall\, a \in A_s : upd(f,x,a) \in \varphi^{\mathcal{A}} \wedge h \circ f =_{\mathit{free}(\varphi)\setminus\{x\}} g \\
&\Rightarrow\; \exists\, f \in (\forall x \varphi)^A : h \circ f =_{\mathit{free}(\forall x \varphi)} g.
\end{aligned}$$

❏

22.3 Restriction with a greatest invariant

Let AX consist of co-Horn clauses $r(t_1,\dots,t_n) \Rightarrow \psi$ and $\mathcal{K} = Alg_{\Sigma,AX}$ such that for all $A \in \mathcal{K}$, $\in^A$ is a Σ-invariant, $t_1,\dots,t_n$ are variables, $free(\psi) \subseteq \{t_1,\dots,t_n\}$ and ψ is $\forall$-free and constraint compatible. Let $C = gfp(\nu\Sigma',\Sigma,AX)$. Then $inv = \in^C$ is the greatest Σ-invariant of $\nu\Sigma'$.

By Lemma 12.7, $inv \in \mathcal{K}$.

Let $A \in \mathcal{K}$. We define $B \in Alg_\Sigma$ as the $unfold^A$-image of the interpretation of R in A, i.e., for all $r \in R$,

$$r^B \ =_{def}\ unfold^A(r^A).$$

Use induction on $\mathbb{N}$ and Theorem 3.4 (or transfinite induction and Theorem 3.8) to show that $unfold^A$ extends to a Σ-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 $unfold^A \circ f\ =_{\{x_1,\dots,x_n\}}\ g$ for some $f \in A^X$. Hence $f \in \psi^A$ because A satisfies φ, and thus by Lemma 22.4 (1),

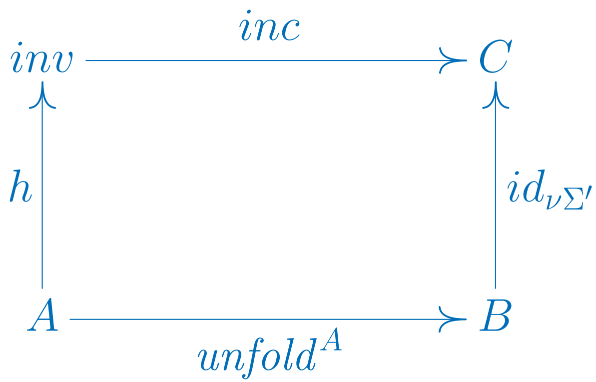
$unfold^A \circ f \in \psi^B$. Therefore, $free(\psi) \subseteq \{x_1, \ldots, x_n\}$ implies $g \in \psi^B$. ❑

Theorem 22.5 *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,

$$img(unfold^A) \;=\; \{unfold^A(a) \mid a \in A\} \;=\; \{unfold^A(a) \mid a \in mem^A\}$$
$$=\; \in^B \;\subseteq\; \in^C \;=\; inv$$

because $\in^A = A$. Hence $h : A \to inv$ is well-defined by $inc \circ h = id_{\nu\Sigma'} \circ unfold^A$.



Since inc is mono and reflects predicates and $id_{\nu\Sigma'} \circ unfold^A$ is Σ-homomorphic, Lemma 9.1 (2) implies that h is also Σ-homomorphic.

Let h' be any Σ-homomorphism from A to inv. Since $B|_{\Sigma'} = \nu\Sigma'$ is final in Alg_Σ, $inc \circ h' = inc \circ h$ and thus $h' = h$ because inc is mono. ❑

Example Length of a colist

Let $\Sigma = (S, F' \cup \{length : list \to nat\}, \{\in : list\})$, $\Sigma' = (S, F' \cup \{length\}, \emptyset)$ and AX be a set of co-Horn clauses such that for all $A \in Alg_{\Sigma,AX}$, p^A_{list} is a Σ-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 22.5, $\in^A$ is final in $Alg_{\Sigma,AX}$. Since the final $coList(X)$-algebra is a (Σ, AX)-algebra, we conclude from Lemma 21.4 that (Σ, AX) is a conservative extension of $(coList(X), \emptyset)$. ❏

Example Subtree of a cobintree

Let $\Sigma = (S, F' \cup \{subtree' : btree \to (2^* \to btree)\}, \{\in : btree\})$,
$\Sigma' = (S, F' \cup \{subtree'\}, \emptyset)$ and AX be a set of co-Horn clauses such that for all $A \in Alg_{\Sigma,AX}$, p^A_{btree} is a Σ-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 22.5, $\in^A$ is final in $Alg_{\Sigma,AX}$. Since the final $coBintree(X)$-algebra is a (Σ, AX)-algebra, we conclude from Lemma 21.4 that (Σ, AX) is a conservative extension of $(coBintree(X), \emptyset)$. ❏

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 Σ-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 $A = lfp(\nu coBintree, \Sigma, AX)$. By Theorem 22.5, $\in^A$ is final in $Alg_{\Sigma,AX}$, the category of Σ-algebras B such that B satisfies AX and $\in^B = B$. ❑

22.4 Restriction with a least invariant

Let AX consist of Horn clauses $r(t_1,\dots,t_n) \Leftarrow \psi$ and $\mathcal{K} = Alg_{\Sigma,AX}$ such that for all $A \in \mathcal{K}$, $\in^A$ is a Σ-invariant, $free(r(t_1,\dots,t_n)) \subseteq free(\psi)$, ψ is constraint compatible and for all atomic subformulas $p(u_1,\dots,u_m)$ of ψ, $u_1,\dots,u_m$ are variables. Let $C = lfp(\nu\Sigma', \Sigma, AX)$ and $inv =\in^C$ be a Σ-invariant of $\nu\Sigma'$. Hence $C \in obs(\mathcal{K})$.

By Lemma 12.7, $inv \in obs(\mathcal{K})$.

Let $A \in \mathcal{K}$. We define $B \in Alg_\Sigma$ as the $unfold^A$-image of the interpretation of R in A, i.e., for all $r \in R$,

$$r^B =_{def} unfold^A(r^A).$$

Use induction on $\mathbb{N}$ and Theorem 3.4 (or transfinite induction and Theorem 3.8) to show that $unfold^A$ extends to a Σ-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 Σ-observable, $unfold^A$ is mono and thus Lemma 22.4 (2) implies $g =_{free(\psi)} unfold^A \circ f$ for some $f \in \psi^A$. Since A satisfies φ, $f \in r(t_1,\dots,t_n)^A$ and thus $(t_1^A(f),\dots,t_n^A(f)) \in r^A$. Hence

$$(t_1^B(unfold^A \circ f),\dots,t_n^B(unfold^A \circ f))$$
$$\stackrel{Lemma\ 9.9}{=} (unfold^A(t_1^A(f)),\dots,unfold^A(t_n^A(f))) \in r^B$$

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 $g \in r(t_1,\dots,t_n)^B$. ❏

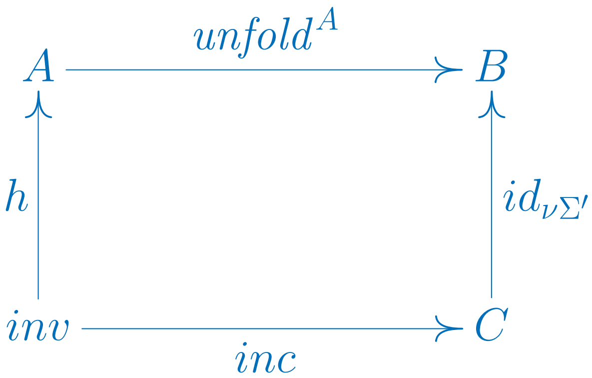
Theorem 22.6 *inv* is initial in $obs(\mathcal{K})$.

Proof. Since C is the least $D \in \mathcal{K}$ with $D|_{\Sigma'} = \nu\Sigma'$, we obtain $C \le B$. In particular,

$$inv \;=\; \in^C \;\subseteq\; \in^B \;=\; \{unfold^A(a) \mid a \in mem^A\} \;=\; \{unfold^A(a) \mid a \in A\}$$
$$=\; img(unfold^A) \qquad (*)$$

because $\in^A = A$. Since A is Σ-observable, $unfold^A$ is mono and thus for all $a,b \in A$, $unfold^A(a) = unfold^A(b)$ implies $a = b$.

Hence by $(*)$, $h : inv \to A$ with $h(b) = (unfold^A)^{-1}(b)$ for all $b \in inv$ is well-defined. Therefore, $unfold^A \circ h = id_{\nu\Sigma'} \circ inc$.



Since $unfold^A$ is mono and reflects predicates and $id_{\nu\Sigma'} \circ inc$ is Σ-homomorphic, Lemma 9.1 (2) implies that h is also Σ-homomorphic.

Let h' be any Σ-homomorphism from inv to A. Since $B|_{B\Sigma} = BA$ is final in Alg_Σ, $unfold^A \circ h' = unfold^A \circ h$ and thus $h' = h$ because $unfold^A$ is mono. ❏

Example Finite trees, EF and AF (see chapter 9)

Let $\Sigma = (S, F', \{finite, EF, AF\})$ and AX be a set of Σ-Horn clauses such that for all $A \in Alg_{\Sigma,AX}$, $\in^A$ is a Σ-invariant. Moreover, let AX include the following axioms:

$$\begin{aligned} \mathit{finite}(t) &\Leftarrow \mathit{split}(t) = \epsilon \vee (\mathit{split}(t) = (x,u,u') \wedge \mathit{finite}(u) \wedge \mathit{finite}(u')) \\ EF(P)(t) &\Leftarrow \mathit{split}(t) = (x,u,u') \wedge (P(x) \vee EF(P)(u) \vee EF(u')) \\ AF(P)(t) &\Leftarrow \mathit{split}(t) = (x,u,u') \wedge (P(x) \vee (AF(P)(u) \wedge AF(u'))) \end{aligned}$$

where P is a predicate variable.

Let $A = \mathit{lfp}(\Sigma, \nu\, \mathit{coBintree}, AX)$. By Theorem 22.6, $\in^A$ is initial in $\mathit{obs}(Alg_{\Sigma,AX})$, the category of F'-observable Σ-coalgebras B such that B satisfies AX and $\in^B = B$. ❏

Example Cotrees with finite outdegree

Let AX be given by the following Horn clauses over coTree:

$$\begin{aligned} is_{tree}(t) &\Leftarrow is_{trees}(subtrees\langle t\rangle) \\ is_{trees}(ts) &\Leftarrow [[x,y]split]ts = [x]p \vee \\ &\quad ([[x,y]split]ts = [y]p \ \wedge is_{tree}(\pi_1\langle p\rangle) \ \wedge is_{trees}(\pi_2\langle p\rangle)) \end{aligned}$$

AX satisfies the assumptions of Restriction with a least invariant (see section 22.4). Hence $inv = \in^{\mathit{lfp}(\overline{AX})}$ is initial in $\mathit{obs}(Alg_{\mathit{coTree},AX})$, the category of coTree-observable coTree-coalgebras A such that A satisfies AX and $\in^A = A$. ❏

23 λ-bialgebras

Given two endofunctors T, D on $\mathcal{K}$, a **distributive law of T over D** is a natural transformation $\lambda : TD \to DT$.

Given a distributive law λ of T over D, a pair of $\mathcal{K}$-morphisms

$$(\alpha : TA \to A,\ \beta : A \to DA),$$

is a λ-**bialgebra** if the following diagram, called **pentagonal law**, commutes:

$$\begin{array}{ccccc}
TA & \xrightarrow{\alpha} & A & \xrightarrow{\beta} & DA \\
{\scriptstyle T\beta}\downarrow & & & & \uparrow{\scriptstyle D\alpha} \\
TDA & & \xrightarrow{\lambda_A} & & DTA
\end{array}$$

λ lifts T and D to endofunctors on $coAlg_T$ and Alg_T, respectively (see, e.g., [26, 91, 73]):

$$\begin{array}{rclcl}
T_\lambda & : & coAlg_D & \to & coAlg_D \\
& & A \xrightarrow{\beta} DA & \mapsto & TA \xrightarrow{T\beta} TDA \xrightarrow{\lambda_A} DTA \\
& & h : A \to B & \mapsto & Th : TA \to TB
\end{array}$$

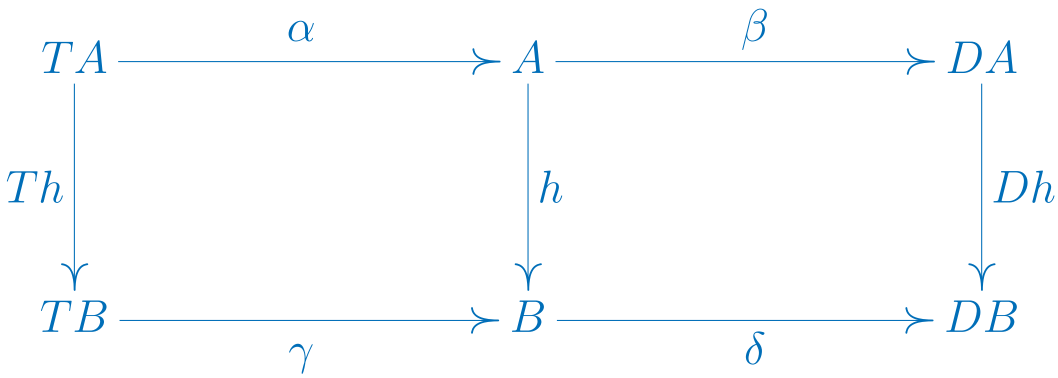
$$
\begin{array}{rcrcl}
D^{\lambda} & : & Alg_T & \to & Alg_T \\
& & TA \xrightarrow{\alpha} A & \mapsto & TDA \xrightarrow{\lambda_A} DTA \xrightarrow{D\alpha} DA \\
& & h : A \to B & \mapsto & Dh : DA \to DB
\end{array}
$$

Hence the pentagonal law can also be written as the following square (1), which shows that α is a $coAlg_D$-morphism from $T_\lambda\beta$ to β, or as the square (2), which shows that β is an Alg_T-morphism from α to $D^\lambda\alpha$:

$$
\begin{array}{ccc}
TA & \xrightarrow{\alpha} & A \\
{\scriptstyle T_\lambda\beta}\downarrow & (1) & \downarrow{\scriptstyle \beta} \\
DTA & \xrightarrow{D\alpha} & DA
\end{array}
\qquad\qquad
\begin{array}{ccc}
A & \xrightarrow{\beta} & DA \\
{\scriptstyle \alpha}\uparrow & (2) & \uparrow{\scriptstyle D^\lambda\alpha} \\
TA & \xrightarrow{T\beta} & TDA
\end{array}
$$

$biAlg_\lambda$ denotes the category of λ-bialgebras and $biAlg_\lambda$-morphisms:

A $biAlg_\lambda$-**morphism** h from a λ-bialgebra $TA \xrightarrow{\alpha} A \xrightarrow{\beta} DA$ to a λ-bialgebra $TB \xrightarrow{\gamma} B \xrightarrow{\delta} DB$ is a $\mathcal{K}$-morphism $h : A \to B$ such that the following diagrams commute: $h \circ \alpha = \gamma \circ Th$ and $Dh \circ \beta = \delta \circ h$.



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 23.1

(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 β of the pentagonal law with $A = \mu T$ agree with $fold^{D^\lambda \alpha}$.

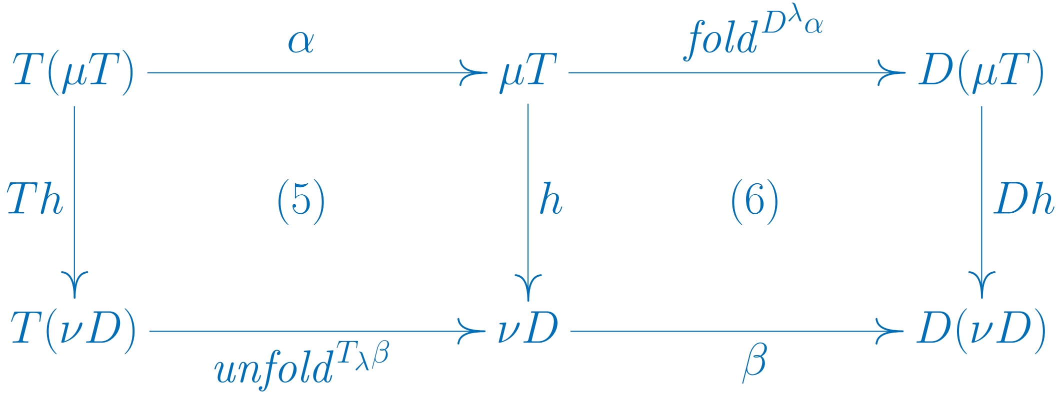
(4) Let $\nu D \xrightarrow{\beta} D(\nu D)$ be final in $coAlg_D$. Then

$$T(\nu D) \xrightarrow{unfold^{T_\lambda \beta}} \nu D \xrightarrow{\beta} D(\nu D)$$

is final in $biAlg_\lambda$. Conversely, all solutions α of the pentagonal law with $A = \nu D$ agree with $unfold^{T_\lambda \beta}$.

Proof. See [91], section 4; or [73], 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 β 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 α is a $coAlg_D$-morphism. ❑

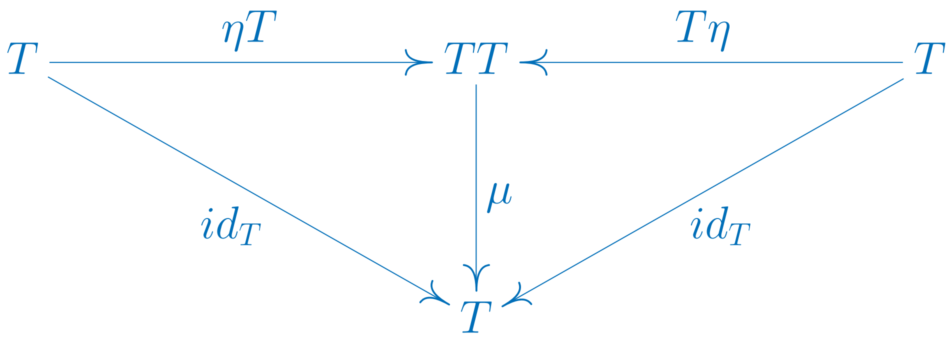
In particular, there is a unique $biAlg_\lambda$-morphism h from the initial λ-bialgebra to the final one:

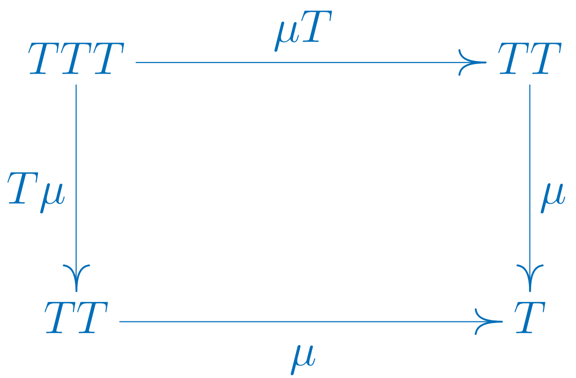


Since $unfold^{T_\lambda \beta}$ equips νD with a T-algebra structure and μT is the initial T-algebra, $h = fold^{\nu D}$. Since $fold^{D^\lambda \alpha}$ equips μT with a D-coalgebra structure and νD is the final D-coalgebra, $h = unfold^{\mu T}$. Hence $fold^{\nu D} = h = unfold^{\mu T}$.

24 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 $T : \mathcal{K} \to \mathcal{K}$ and natural transformations $\eta : Id_{\mathcal{K}} \to T$ (**unit**) and $\mu : TT \to T$ (**multiplication**) such that the following diagrams commute:





If η and μ are clear from the context, M is abbreviated to 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 $T : \mathcal{K} \to \mathcal{K}$, sets

$$\eta = \{\eta_A : A \to TA \mid A \in \mathcal{K}\} \quad \text{and} \quad \_^* = \{f^* : TA \to TB \mid A, B \in \mathcal{K},\ f : A \to TB\}$$

such that for all $A \in \mathcal{K}$, $f : A \to TB$ and $g : B \to TC$,

$$\eta_A^* = id_{TA}, \quad f^* \circ \eta_A = f, \quad (g^* \circ f)^* = g^* \circ f^*.$$

A Kleisli triple $(T, \eta, \_^*)$ defines the monad (T, η, μ) with $Tf = (\eta_B \circ f)^*$ and $\mu_A = id_{TA}^*$ for all $A \in \mathcal{K}$ and $f : A \to B$.

Conversely, a monad (T, η, μ) defines the Kleisli triple $(T, \eta, \_^*)$ with $f^* = \mu_B \circ Tf$ for all $f : A \to TB$.

Haskell implements $\_^*$ by the **bind operator**

$$bind = (>>=) : TA \to (A \to TB) \to TB :$$

$bind(t)(f) =_{def} f^*(t)$.

In Haskell, η and μ are called *return* and *join*, respectively (see here).

The **Kleisli composition**

$$\circ_T : (B \to TC) \times (A \to TB) \to (A \to TC)$$

combines *bind* with function composition: $g \circ_T f =_{def} g^* \circ f$.

η 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 $\eta_A : A \to TA$.

24.1 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:

$$\begin{aligned} \eta_A : A &\to A^* \\ a &\mapsto (a) \end{aligned} \qquad \begin{aligned} \mu_A : (A^*)^* &\to A^* \\ (w_1, \ldots, w_n) &\mapsto w_1 \cdot \ldots \cdot w_n \end{aligned} \qquad \begin{aligned} bind : A^* &\to (A \to B^*) \to B^* \\ s &\mapsto \lambda f.\mu_B(map(f)(s)) \end{aligned}$$

- powerset functor:

$$\begin{aligned} \eta_A : A &\to \mathcal{P}(A) \\ a &\mapsto \{a\} \end{aligned} \qquad \begin{aligned} \mu_A : \mathcal{P}(\mathcal{P}(A)) &\to \mathcal{P}(A) \\ S &\mapsto \bigcup S \end{aligned}$$

$$\begin{aligned} bind : \mathcal{P}(A) &\to (A \to \mathcal{P}(B)) \to \mathcal{P}(B) \\ S &\to \lambda f.\bigcup\{f(s) \mid s \in S\} \end{aligned}$$

$$\begin{aligned} \circ_{\mathcal{P}} : (B \to \mathcal{P}(C)) \times (A \to \mathcal{P}(B)) &\to (A \to \mathcal{P}(C)) \\ (g, f) &\mapsto \lambda a.\bigcup\{g(b) \mid b \in f(a)\} \end{aligned}$$

- finite-set functor $\mathcal{P}_\omega$: analogously

- M-weighted-set functor: Let $(M, +, 0, *, 1)$ be a semiring.

$$\begin{array}{rcl} \eta_A : A & \to & M_\omega^A \\ a & \mapsto & (\lambda x.0)[1/a] \end{array} \qquad \begin{array}{rcl} \mu_A : M_\omega^{M_\omega^A} & \to & M_\omega^A \\ g & \mapsto & \lambda a. \sum\{g(h) * h(a) \mid h \in supp(g)\} \end{array}$$

$$\begin{array}{rcl} bind : M_\omega^A & \to & (A \to M_\omega^B) \to M_\omega^A \\ h & \mapsto & \lambda f.\lambda b. \sum\{h(a) * f(a)(b) \mid a \in supp(g)\} \end{array}$$

Kleisli composition is matrix multiplication:

$$\begin{array}{rcl} \circ_{M_\omega^-} : (B \to M_\omega^C) \times (A \to M_\omega^B) & \to & (A \to M_\omega^C) \\ (g, f) & \mapsto & \lambda a. \sum\{g(b) * f(a) \mid b \in supp(g)\} \end{array}$$

- distribution functor:

$$\begin{array}{rcl} \eta_A : A & \to & \mathcal{D}(A) \\ a & \mapsto & (\lambda x.0)[1/a] \end{array} \qquad \begin{array}{rcl} \mu_A : \mathcal{D}(\mathcal{D}(A)) & \to & \mathcal{D}(A) \\ g & \mapsto & \lambda a. \sum\{g(h) * h(a) \mid h \in supp(g)\} \end{array}$$

$$\begin{array}{rcl} bind : \mathcal{D}(A) & \to & (A \to \mathcal{D}(B)) \to \mathcal{D}(B) \\ h & \mapsto & \lambda f.\lambda b. \sum\{h(a) * f(a)(b) \mid a \in supp(g)\} \end{array}$$

- exception functor:

$$\begin{array}{rcl} \eta_A : A & \to & A + X \\ a & \mapsto & (a, 1) \end{array} \qquad \begin{array}{rcl} \mu_A : (A + X) + X & \to & A + X \\ ((a,1),1) & \mapsto & (a,1) \\ ((x,2),1) & \mapsto & (x,2) \\ (x,2) & \mapsto & (x,2) \end{array}$$

If $X = 1$, then Kleisli composition is composition of partial functions:

$$\begin{array}{rcl} \circ_{\_+1} : (B \to C + 1) & \times & (A \to B + 1) \to (A \to C + 1) \\ g & \mapsto & \lambda f.\lambda a. \begin{cases} (g(f(b)), 1) & \text{if } f(a) = (b, 1) \text{ for some } b \in B \\ (\epsilon, 2) & \text{if } f(a) = (\epsilon, 2) \end{cases} \end{array}$$

- reader functor:

$$\begin{array}{rcl} \eta_A : A & \to & A^X \\ a & \mapsto & \lambda x.a \end{array} \qquad \begin{array}{rcl} \mu_A : (A^X)^X & \to & A^X \\ g & \mapsto & \lambda x.g(x)(x) \end{array} \qquad \begin{array}{rcl} bind : A^X & \to & (A \to B^X) \to B^X \\ h & \mapsto & \lambda f.\lambda x.f(h(x))(x) \end{array}$$

- writer functor: Let $(X, +, 0)$ be a monoid.

$$\begin{array}{rcl} \eta_A : A & \to & A \times X \\ a & \mapsto & (a, 0) \end{array} \qquad \begin{array}{rcl} \mu_A : (A \times X) \times X & \to & A \times X \\ ((a, x), y) & \mapsto & (a, x + y) \end{array}$$

$$\begin{array}{rcl} \mathit{bind} : A \times X & \to & (A \to B \times X) \to B \times X \\ (a, x) & \mapsto & \lambda f.(\lambda(b, y).(b, x + y))(f(a)) \end{array}$$

- state functor:

$$\begin{array}{rcl} \eta_A : A & \to & (A \times X)^X \\ a & \mapsto & \lambda x.(a, x) \end{array} \qquad \begin{array}{rcl} \mu_A : ((A \times X)^X \times X)^X & \to & (A \times X)^X \\ g & \mapsto & \mathit{apply} \circ g \end{array}$$

$$\begin{array}{rcl} \mathit{bind} : (A \times X)^X & \to & (A \to (B \times X)^X) \to (B \times X)^X \\ h & \mapsto & \lambda f.\mathit{uncurry}(f) \circ h \end{array}$$

For the state functor $T = (\_ \times X)^X$ and $f : A \to TB$, the equation $f^* = \mu_B \circ Tf$ (see above) entails the following definition of $f^* : TA \to TB$:

For all $g : X \to A \times X$ and $x \in X$,

$$f^*(g)(x) = f(\pi_1(gx))(\pi_2(gx)).$$

Let $(T : Set \to Set, \eta, \mu)$ 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 $f : A \to B$ to the function

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

$$\begin{aligned} \eta'_A : A &\to T(A \times X)^X \\ a &\mapsto \lambda x.\eta(a, x) \end{aligned} \qquad \begin{aligned} \mu'_A : (T(A \times X)^X \times X)^X &\to T(A \times X)^X \\ g &\mapsto (\lambda(h, x).h(x)) \circ_T g \end{aligned}$$

$$\begin{aligned} bind' : T(A \times X)^X &\to (A \to T(B \times X)^X) \to T(B \times X)^X \\ h &\mapsto \lambda f.uncurry(f) \circ_T h \end{aligned}$$

Let (L, R, ϕ, ψ) be an adjunction from $\mathcal{K}$ to $\mathcal{L}$ with unit η and co-unit ϵ (see section 19.1).

$M(L, R, \phi, \psi) = (RL, \eta, R\epsilon L : RLRL \to RL)$ is a monad, called **the monad induced by** (L, R, ϕ, ψ).

The monoid monad

The list functor $\_^* : Set \to Set$ coincides with UMF and thus $(\_^*, \eta, U\epsilon MF)$ is the monad induced by the adjunction of section 19.3.

The sequence monad

Given the adjunction of section 19.4, the writer functor $X^* \times \_ : Set \to Set$ coincides with USF_X and thus $(X^* \times \_, \eta, U\epsilon SF_X)$ is the monad induced by this adjunction.

24.2 Term monads

Let $\Sigma = (S, C)$ be a constructive polynomial signature.

The monad induced by the adjunction adj_Σ of section 19.12 is called the **monad freely generated by** Σ (see).

The categories $Alg_{M(adj_\Sigma)}$ and Alg_Σ are isomorphic.

Let $V, V' \in Set_b^S$ (see chapter 7) and $g \in T_\Sigma(V')^V$.

The equation $bind(t)(g) = g^*(t)$ (see above) yields the following definition of

$$bind : T_\Sigma(V) \to (V \to T_\Sigma(V')) \to T_\Sigma(V').$$

- For all $s \in S$ and $x \in V_s$, $bind(x)(g) = g(x)$.
- For all $c : e \to s \in C$ and $t \in T_\Sigma(V)_e$, $bind(c(t))(g) = c^{\mathcal{A}}(bind(t)(g))$.
- For all $e = \prod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $t = (t_i)_{i \in I} \in \times_{i \in I} T_\Sigma(V)_{e_i}$ and $i \in I$,

$$\pi_i(bind(t)(g)) = bind(t_i)(g).$$

- For all $e = \coprod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $t \in T_\Sigma(V)_{e_i}$,

$$bind(i(t))(g) = i(bind(t)(g)).$$

Hence for all $t \in T_\Sigma(V)$ and $g : V \to T_\Sigma(V')$, $bind(t)(g) \in T_\Sigma(V')$ 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 $\mu : TT \to T$:

- For all $s \in S$ and $t \in T_\Sigma(V)_s$, $\mu_V(t) = t$.
- For all $c : e \to s \in C$ and $t \in T_\Sigma(T_\Sigma(V))_e$,

$$\mu_V(c(t)) = c(\mu_V(t)).$$

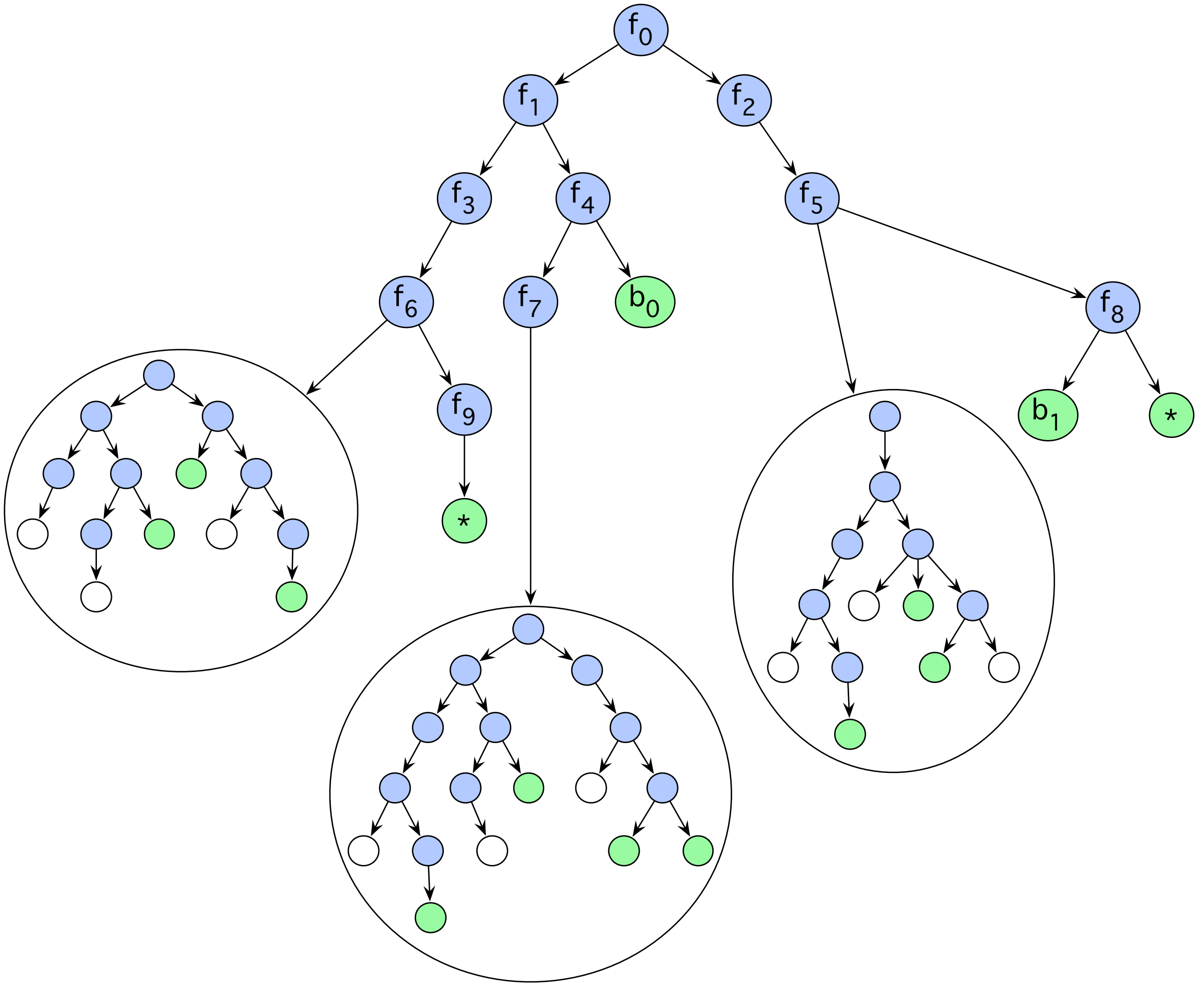
- For all $e = \prod_{i \in I} e_i \in \mathcal{T}_{po}(S)$ and $t = (t_i)_{i \in I} \in \times_{i \in I} T_\Sigma(T_\Sigma(V))_{e_i}$ and $i \in I$,

$$\pi_i(\mu_V(t)) = \mu_V(t_i).$$

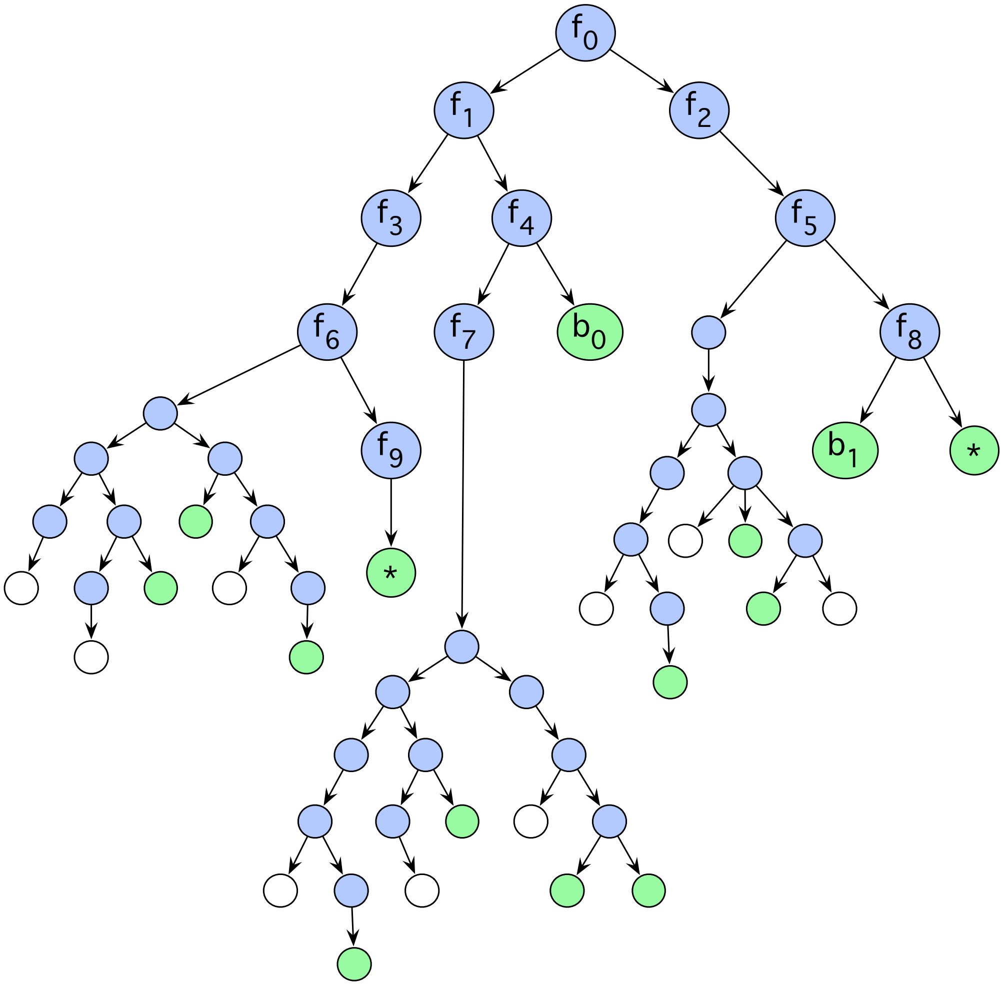
- For all $e = \coprod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $t \in T_\Sigma(T_\Sigma(V))_{e_i}$,

$$\mu_V(i(t)) = i(\mu_V(t)).$$

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.



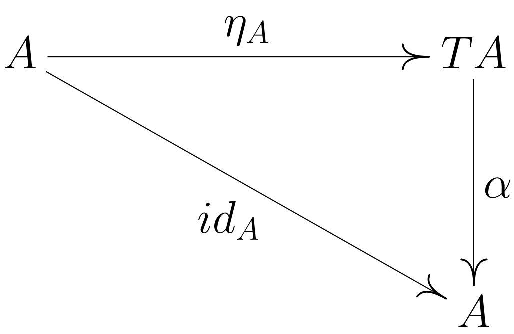
A term t over $T_\Sigma(V)$

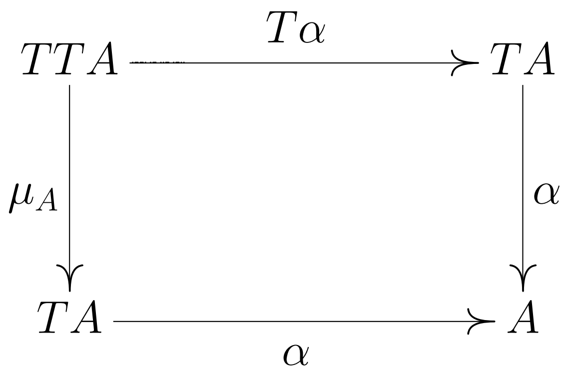


The term over V *that results from applying* $\mu_V : T_\Sigma(T_\Sigma(V)) \to T_\Sigma(V)$ *to* t

Let $M = (T : \mathcal{K} \to \mathcal{K}, \eta, \mu)$ be a monad.

An M**-algebra** or **Eilenberg-Moore algebra** is a T-algebra $\alpha : TA \to A$ such that the following diagrams commute:





The category of M-algebras is denoted by Alg_M. Alg_M is a full subcategory of Alg_T.

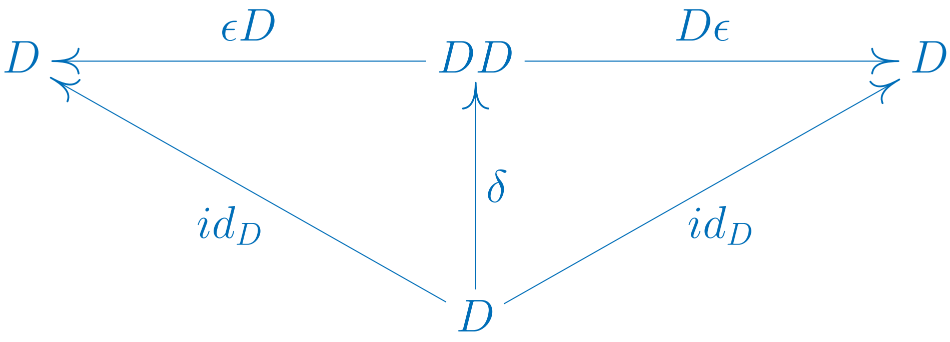
The forgetful functor $U : Alg_M \to \mathcal{K}$ has a left adjoint $L : \mathcal{K} \to Alg_M$.

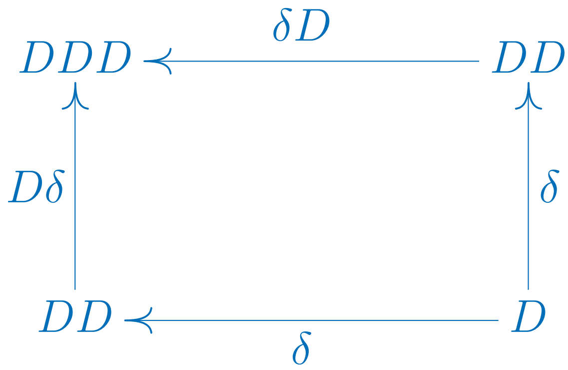
Let $adj_M = (L, U, \phi, \psi)$ be the corresponding adjunction.

The monad induced by adj_M coincides with M: $M(adj_M) = M$.

Let $\Sigma = (S, C)$ be a constructive polynomial signature and $\mathcal{A}$ be a Σ-algebra with carrier A. Then $id_A^* : T_\Sigma(A) \to A$ is an Eilenberg-Moore T_Σ-algebra.

A **comonad** in $\mathcal{K}$ is a triple $CM = (D, \epsilon, \delta)$ consisting of a functor $D : \mathcal{K} \to \mathcal{K}$ and natural transformations $\epsilon : D \to Id_{\mathcal{K}}$ (**co-unit**) and $\delta : D \to DD$ (**comultiplication**) such that the following diagrams commute:





A **co-Kleisli triple** $(D, \epsilon, \_^{\#})$ consists of a function $D : \mathcal{K} \to \mathcal{K}$, sets

$\epsilon = \{\epsilon_A : DA \to A \mid A \in \mathcal{K}\}$ and $\_^{\#} = \{f^{\#} : DA \to DB \mid A, B \in \mathcal{K},\ f : DA \to B\}$

such that for all $A \in \mathcal{K}$, $f : DA \to B$ and $g : DB \to C$,

$$\epsilon_A^{\#} = id_{DA}, \quad \epsilon_B \circ f^{\#} = f, \quad (g \circ f^{\#})^{\#} = g^{\#} \circ f^{\#}.$$

A co-Kleisli triple $(D, \epsilon, \_^{\#})$ defines the comonad (D, ϵ, δ) with $Df = (f \circ \epsilon_A)^{\#}$ and $\delta_A = id_{DA}^{\#}$ for all $A \in \mathcal{K}$ and $f : A \to B$.

Conversely, a comonad (D, ϵ, δ) defines the co-Kleisli triple $(D, \epsilon, \_^{\#})$ with $f^{\#} = Df \circ \delta_A$ for all $f : DA \to B$.

Haskell implements $\_^{\#}$ by the **cobind operator**

$$cobind = (<\!\!<\!=) : (DA \to B) \to (DA \to DB) :$$

$cobind(f)(d) =_{def} f^{\#}(d)$.

In Haskell, ϵ and δ are called *retract* and *duplicate*, respectively (see here).

The **co-Kleisli composition**

$$\circ^D = (=\!>\!=) : (DB \to C) \times (DA \to B) \to (DA \to C)$$

combines *cobind* with function composition: $g \circ^D f =_{def} g \circ f^{\#}$.

ϵ 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 $\epsilon_A : DA \to A$.

24.3 Sample comonads

Many functors defined in chapter 5 are also comonads. Co-unit, comultiplication and cobind are defined as follows:

- identity functor: $\epsilon_A = \delta_A = id_A$. $cobind = id_{A \to B}$.
- list functor:

$$\begin{array}{rcl} \epsilon_A : A^+ & \to & A \\ (a_1, \ldots, a_n) & \mapsto & a_1 \end{array} \qquad \begin{array}{rcl} \delta_A : A^+ & \to & (A^+)^+ \\ (a_1, \ldots, a_n) & \mapsto & ((a_1, \ldots, a_n), (a_2, \ldots, a_n), \ldots, a_n) \end{array}$$

$$\begin{array}{rcl} cobind : (A^+ \to B) & \to & (A^+ \to B^+) \\ f & \mapsto & \lambda(a_1, \ldots, a_n).(f(a_1, \ldots, a_n), f(a_2, \ldots, a_n), \ldots, f(a_n)) \end{array}$$

- reader functor: Let $(X, +, 0)$ be a monoid.

$$\begin{array}{rcl} \epsilon_A : A^X & \to & A \\ h & \mapsto & h(0) \end{array} \qquad \begin{array}{rcl} \delta_A : A^X & \to & (A^X)^X \\ h & \mapsto & \lambda x.\lambda y.h(x + y) \end{array}$$

$$\begin{array}{rcl} cobind : (A^X \to B) & \to & (A^X \to B^X) \\ f & \mapsto & \lambda h.\lambda x.f(\lambda y.h(x + y)) \end{array}$$

- writer functor:

$$\begin{array}{rcl} \epsilon_A : A \times X & \rightarrow & A \\ (a,x) & \mapsto & a \end{array} \qquad \begin{array}{rcl} \delta_A : A \times X & \rightarrow & (A \times X) \times X \\ (a,x) & \mapsto & ((a,x),x) \end{array}$$

$$\begin{array}{rcl} \mathit{cobind} : (A \times X \rightarrow B) & \rightarrow & (A \times X \rightarrow B \times X) \\ f & \mapsto & \lambda(a,x).(f(a,x),x) \end{array}$$

- costate functor:

$$\begin{array}{rcl} \epsilon_A : A^X \times X & \rightarrow & A \\ (h,x) & \mapsto & h(x) \end{array} \qquad \begin{array}{rcl} \delta_A : A^X \times X & \rightarrow & (A^X \times X)^X \times X \\ (h,x) & \mapsto & (\lambda y.(h,y),x) \end{array}$$

$$\begin{array}{rcl} \mathit{cobind} : (A^X \times X \rightarrow B) & \rightarrow & (A^X \times X \rightarrow B^X \times X) \\ f & \mapsto & \lambda(h,x).(\lambda y.f(h,y),x) \end{array}$$

- labelled-tree functor:

$$\begin{array}{rcl} \epsilon_A : \mathit{ltr}(X,A) & \rightarrow & A \\ t & \mapsto & t(\epsilon) \end{array} \qquad \begin{array}{rcl} \delta_A : \mathit{ltr}(X,A) & \rightarrow & \mathit{ltr}(X,\mathit{ltr}(X,A)) \\ t & \mapsto & \lambda v.\lambda w.t(vw) \end{array}$$

$$\begin{array}{rcl} \mathit{cobind} : (\mathit{ltr}(X,A) \rightarrow B) & \rightarrow & (\mathit{ltr}(X,A) \rightarrow \mathit{ltr}(X,B)) \\ f & \mapsto & \lambda t.\lambda v.f(\lambda w.t(vw)) \end{array}$$

- pointed-tree functor:

$$\begin{aligned} \epsilon_A : ltr(X,A) \times X^* &\to A \\ (t,w) &\mapsto t(w) \end{aligned}$$

$$\begin{aligned} \delta_A : ltr(X,A) \times X^* &\to ltr(X, ltr(X,A) \times X^*) \times X^* \\ (t,v) &\mapsto (\lambda w.(t,w), v) \end{aligned}$$

$$\begin{aligned} cobind : (ltr(X,A) \times X^* \to B) &\to (ltr(X,A) \times X^* \to ltr(X,B) \times X^*) \\ f &\mapsto \lambda(t,v).(\lambda w.f(t,w), v) \end{aligned}$$

Let (L, R, ϕ, ψ) be an adjunction from $\mathcal{K}$ to $\mathcal{L}$ with unit η and co-unit ϵ (see section 19.1).

$CM(L, R, \phi, \psi) = (LR, \epsilon, L\eta R : LR \to LRLR)$ is a comonad, called **the comonad induced by** (L, R, ϕ, ψ).

The behavior comonad

Let X be a set. Given the adjunction of section 19.5, the reader functor $\_^{X^*} : Set \to Set$ coincides with UBF_X and thus $(\_^{X^*}, \epsilon, U\eta BF_X)$ is the comonad induced by this adjunction.

24.4 Coterm comonads

Let $\Sigma = (S, D)$ be a destructive polynomial signature and $B = \bigcup \mathcal{I}$.

The comonad induced by the adjunction adj_Σ of section 19.15 is called the **comonad cofreely generated by** Σ.

The categories $coAlg_{CM(adj_\Sigma)}$ and $coAlg_\Sigma$ are isomorphic.

Let C, C' be as in section 19.15 and $g \in C'^{DT_\Sigma(C)}$.

The equation $cobind(g)(t) = g^\#(t)$ (see above) yields the following definition of

$$cobind : (DT_\Sigma(C) \to C') \to DT_\Sigma(C) \to DT_\Sigma(C') :$$

- For all $s \in S$ and $t \in DT_\Sigma(C)_s$,

$$cobind(g)(t) = g(t)\{d \to cobind(g)(d^{\mathcal{A}}(t)) \mid d : s \to e \in D\}.$$

- For all $e = \prod_{i \in I} e_i \in \mathcal{T}_{po}(S)$ and $t = (t_i)_{i \in I} \in \times_{i \in I} DT_\Sigma(C)_{e_i}$,

$$\pi_i(cobind(g)(t)) = cobind(g)(t_i).$$

- For all $e = \coprod_{i \in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $t \in DT_\Sigma(C)_{e_i}$,

$$cobind(g)(i(t)) = i(cobind(g)(t)).$$

Hence for all $g : DT_\Sigma(C) \to C'$ 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 $\delta : D \to DD$:

- For all $s \in S$ and $t \in DT_\Sigma(C)_s$,

$$\delta_C(t) = t\{d \to \delta_C(t_d)(w) \mid d : s \to e \in D\}.$$

- For all $e = \prod_{i\in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $t = (t_i)_{i\in I} \in \times_{i\in I} DT_\Sigma(C)_{e_i}$,

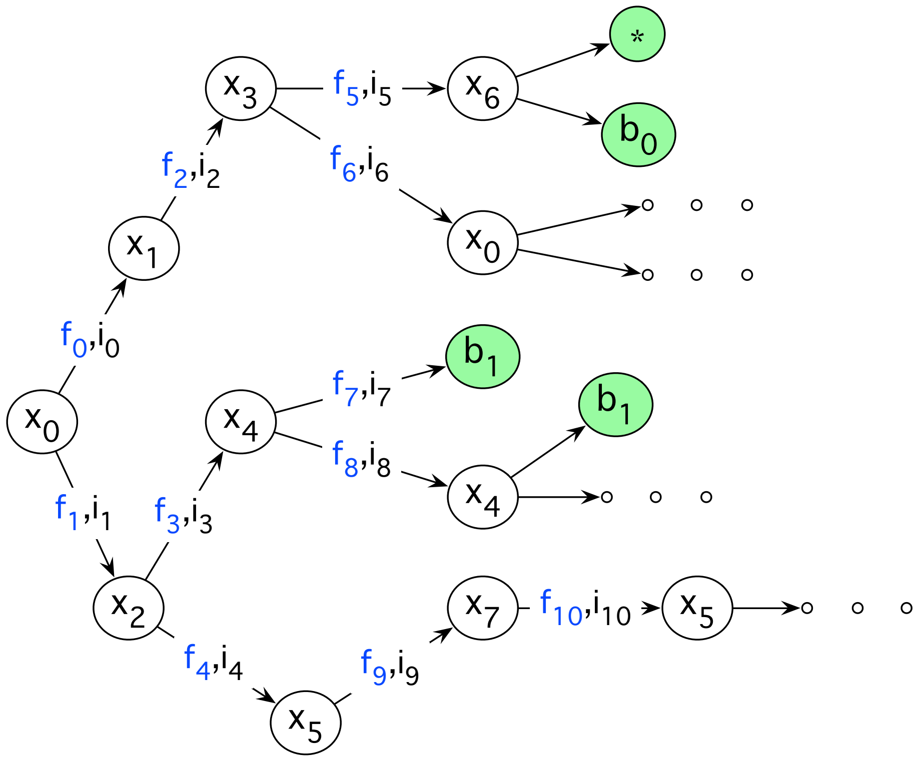
$$\pi_i(\delta_C(t)) = \delta_C(t_i).$$

- For all $e = \coprod_{i\in I} e_i \in \mathcal{T}_{po}(S)$, $i \in I$ and $t \in DT_\Sigma(C)_{e_i}$, $\delta_C(i(t)) = i(\delta_C(t))$.

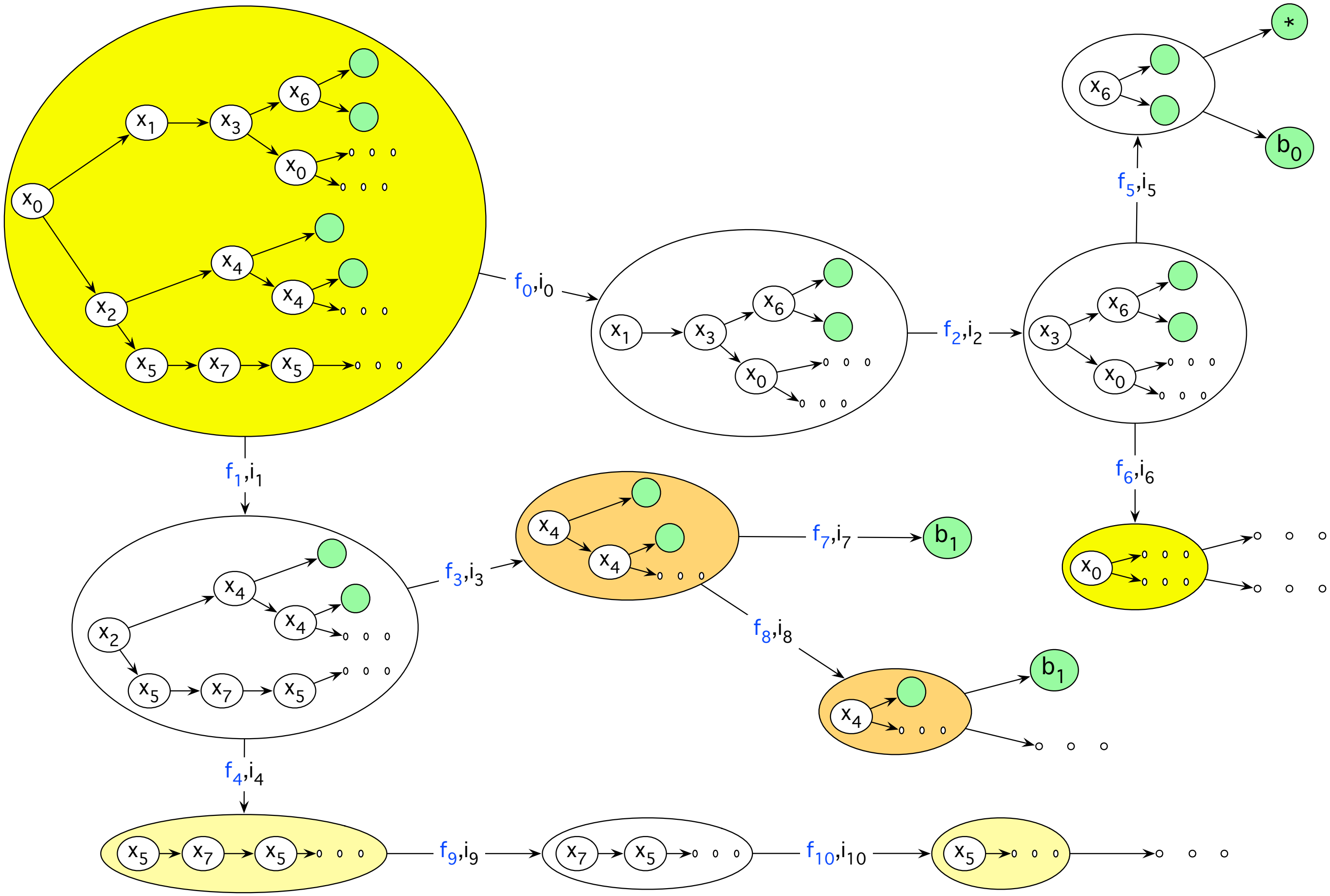
For all $t \in DT_\Sigma(C)$, $\delta_C(t) \in DT_\Sigma(DT_\Sigma(C))$ is obtained from t by replacing $t(w)$ with the subtree of t with root position w, i.e.,

$$\delta_C(t)(w) = \lambda v.t(wv)$$

(see section 9.3).



A coterm t over C

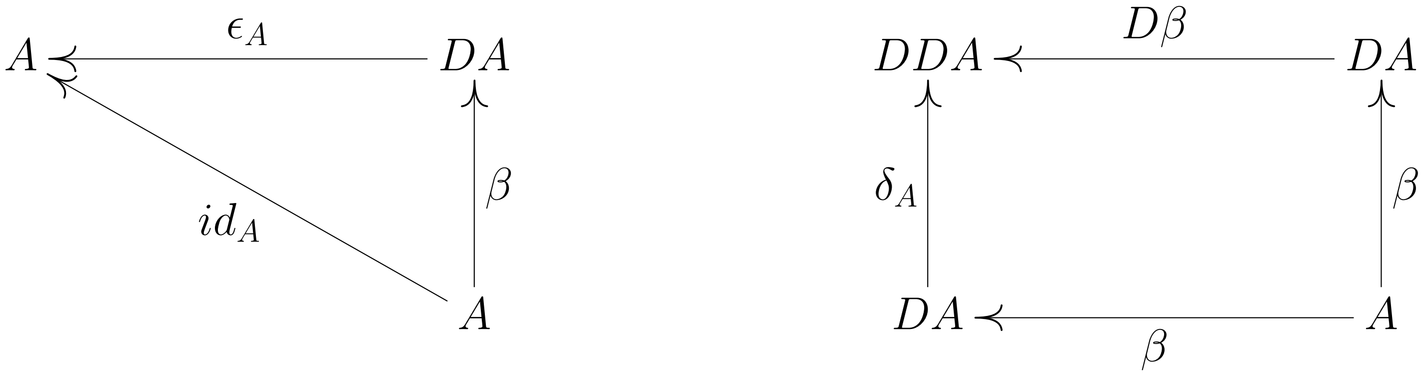


The coterm over $DT_\Sigma(C)$ that results

from applying $\delta_V : DT_\Sigma(C) \to DT_\Sigma(DT_\Sigma(C))$ to t

Let $CM = (D : \mathcal{K} \to \mathcal{K}, \epsilon, \delta)$ be a comonad.

A CM**-coalgebra** or **Eilenberg-Moore coalgebra** is a D-coalgebra $\beta : A \to DA$ such that the following diagrams commute:



The category of CM-coalgebras is denoted by $coAlg_{CM}$. $coAlg_{CM}$ is a full subcategory of $coAlg_D$.

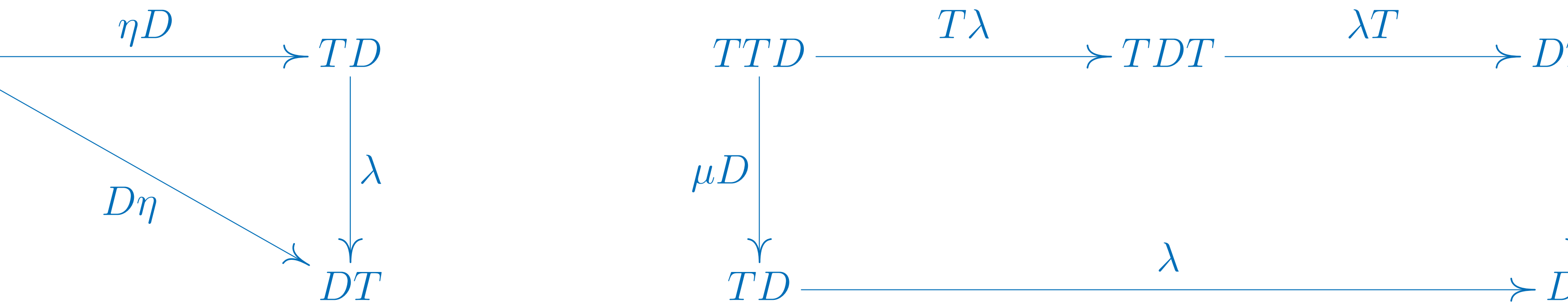
The forgetful functor $U : coAlg_{CM} \to \mathcal{K}$ has a right adjoint $R : \mathcal{K} \to coAlg_{CM}$.

Let $adj_{CM} = (U, R, \phi, \psi)$ be the corresponding adjunction.

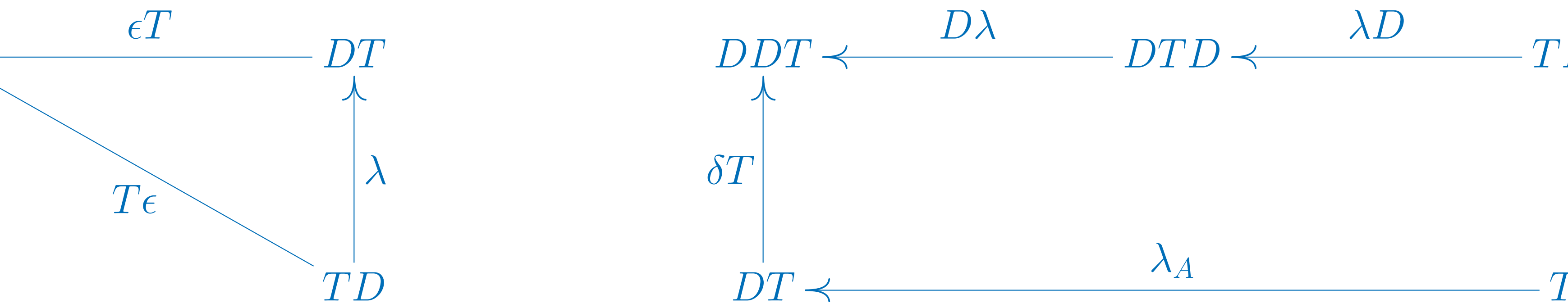
The comonad induced by adj_{CM} coincides with CM: $CM(adj_{CM}) = CM$.

Let $\Sigma = (S, D)$ be a destructive polynomial signature and $\mathcal{A}$ be a Σ-algebra with carrier A. Then $id_A^{\#} : A \to DT_\Sigma(A)$ is an Eilenberg-Moore DT_Σ-coalgebra.

Given a monad $M = (T, \eta, \mu)$, a distributive law $\lambda : TD \to DT$ is M-**compatible** if the following diagrams commute:



Given a comonad $CM = (D, \epsilon, \delta)$, a distributive law $\lambda : TD \to DT$ is CM-**compatible** if the following diagrams commute:



Examples

Given a monad $M = (T, \eta, \mu)$ in *Set*, the strength $st^{T,A}$ of T and A is M-compatible (see Example 3 in section 5.1).

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 $\alpha : TB \to B$, let $D = (-)^A \times B$.

$$\lambda : TD \to DT$$

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. ❏

Theorem 25.1 (inspired by [72])

Let $\Sigma = (S, C)$ be a constructive signature, (L, R, ϕ, ψ) be an adjunction from $Mod(S)$ to $\mathcal{K}$, $\mathcal{A}$ be an initial Σ-algebra with carrier A and $B \in \mathcal{K}$ such that $R(B)$ is a Σ-algebra.

There is a unique $\mathcal{K}$-morphism $d : LA \to B$ such that for all $c : e \to s \in C$,

$$d_s^\# \circ c^{\mathcal{A}} = c^{R(B)} \circ d_e^\#. \tag{1}$$

d is (L, R)**-recursive** if (1) holds true for all $c \in C$.

Proof. Let $d = \epsilon_B \circ L(fold^{RB})$. There is a unique S-sorted function $d^\# : A \to RB$ such that $\epsilon_B \circ L(d^\#) = d$. Hence $d^\# = fold^{RB}$ and thus $d^\#$ is Σ-homomorphic, i.e., (1) holds true.

Let $f, g : LA \to B$ be two $\mathcal{K}$-morphisms such that $f^\#$ and $g^\#$ are Σ-homomorphic. Since $\mathcal{A}$ is initial in Alg_Σ, $f^\# = fold^{RB} = g^\#$. Hence $f = \epsilon_B \circ L(f^\#) = \epsilon_B \circ L(g^\#) = g$. ❑

Like Theorem 16.1, Theorem 25.1 covers the simultaneous definition of the elements of a function tuple $(f_i : A \to B_i)_{i\in I}$. Provided that S is a singleton,

$$(\Delta^I : Set \to Set^I,\ \prod_{i\in I} : Set^I \to Set,\ \lambda f.(\pi_i \circ f)_{i\in I},\ \lambda(f_i)_{i\in I}.\langle f_i\rangle_{i\in I})$$

is the appropriate adjunction (see section 19.11). Theorem 25.1 (1) can be turned into Theorem 16.1 (1).

In the following example, Theorem 25.1 is applied to the reader-writer adjunction (see section 19.9) in order to define a *binary* function inductively.

Example

Let $\Sigma = Nat$, $L = \_ \times \mathbb{N}$ and $R = \_^{\mathbb{N}}$. Then $L(\mathbb{N}) = \mathbb{N}^2$ $R(\mathbb{N}) = \mathbb{N}^{\mathbb{N}}$ and we interpret the arrows $zero : 1 \to nat$ and $succ : nat \to nat$ 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, $add : L(\mathbb{N}) \to \mathbb{N}$, 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 $add^{\#} = curry(add) : \mathbb{N} \to R(\mathbb{N})$ satisfies (1), i.e.,

$$add^{\#}(0) = zero^{R(\mathbb{N})}, \tag{4}$$
$$add^{\#}(m+1) = succ^{R(\mathbb{N})}(add^{\#}(m)) \tag{5}$$

for all $m \in \mathbb{N}$.

Proof. "⇒": Let (2) and (3) hold true. Then

$$add^{\#}(0) = curry(add)(0) = (\lambda m.\lambda n.add(m,n))(0) = \lambda n.add(0,n) \overset{(2)}{=} \lambda n.n = zero^{R(\mathbb{N})},$$

and for all $m \in \mathbb{N}$,

$$add^{\#}(m+1) = curry(add)(m+1) = (\lambda m'.\lambda n.add(m',n))(m+1) = \lambda n'.add(m+1,n)$$
$$\overset{(3)}{=} add(m,n)+1 = d^{\#}(m)(n)+1 = (\lambda f.\lambda n.f(n)+1)(d^{\#}(m)) = succ^{R(\mathbb{N})}(d^{\#}(m)),$$

i.e., (4) and (5) are valid.

"⇐": Let (4) and (5) hold true. Then (2) and (3) follow from a rearrangement of the preceding equations.

Hence by Theorem 25.1, *add* is uniquely defined by (2) and (3). ❑

Exercise 21 Let Σ, L, R and the Σ-algebra $R(\mathbb{N})$ be defined as in the previous example. Show the following equivalence:

The multiplication of natural numbers, $mult : L(\mathbb{N}) \to \mathbb{N}$, 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 $mult^{\#} = curry(mult) : \mathbb{N} \to R(\mathbb{N})$ satisfies (4) and (5) with *mult* instead of *add*.

Hence by Theorem 25.1, *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., sample inductive definition 16.3.8).

Theorem 25.2 (inspired by [72])

Let $\Sigma = (S, D)$ be a destructive signature,

$$(L : \mathcal{K} \to Mod(S), R : Mod(S) \to \mathcal{K}, \phi, \psi)$$

be an adjunction, $\mathcal{A}$ be a final Σ-algebra with carrier A and $B \in \mathcal{K}$ such that $L(B)$ is a Σ-algebra.

There is a unique $\mathcal{K}$-morphism $c : B \to R(A)$ such that for all $d : s \to e \in D$,

$$d^{\mathcal{A}} \circ c_s^* = c_e^* \circ d^{\mathcal{A}}. \tag{2}$$

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, η, ϵ) is an adjunction, there is a unique S-sorted function $c^* : L(B) \to A$ such that $R(c^*) \circ \eta_B = c$. Hence $c^* = unfold^{L(B)}$ and thus c^* is Σ-homomorphic.

Let $f, g : B \to R(A)$ be two $\mathcal{K}$-morphisms such that f^* and g^* are Σ-homomorphic. Since $\mathcal{A}$ is final in Alg_Σ, $f^* = unfold^{L(B)} = g^*$. Hence $f = R(f^*) \circ \eta_B = R(g^*) \circ \eta_B = g$. ❏

Like Theorem 16.2, Theorem 25.2 covers the simultaneous definition of the elements of a function tuple $(f_i : B_i \to A)_{i\in I}$. Provided that S is a singleton,

$$(\coprod_{i\in I} : Set^I \to Set,\ \Delta^I : Set \to Set^I,\ \lambda(f_i)_{i\in I}.[f_i]_{i\in I},\ \lambda f.(f \circ \iota_i)_{i\in I}$$

is the appropriate adjunction (see section 19.11). Theorem 25.2 (1) can be turned into Theorem 16.2 (1).

Given a constructive or destructive signature Σ, Lemma 23.1 suggests to define operations on the initial or final Σ-algebra in terms of distributive laws of or over H_Σ, respectively:

Theorem 25.3

Let $\Sigma = (S, F)$ be a constructive signature and $\lambda : H_\Sigma D \to DH_\Sigma$ be a distributive law of H_Σ over some endofunctor D on Set^S.

Let $\mu\Sigma$ be initial in Alg_Σ. Then $\alpha : H_\Sigma(\mu\Sigma) \to \mu\Sigma$ with $H_\Sigma(\mu\Sigma)_s = \coprod_{c:e\to s\in F} \mu\Sigma_e$ and $\alpha_s = [c^{\mu\Sigma}]_{c:e\to s\in F}$ for all $s \in S$ is initial in Alg_{H_Σ} (see chapter 15).

An S-sorted function $f : \mu\Sigma \to D(\mu\Sigma)$ is **λ-recursive** if (α, f) is a λ-bialgebra, i.e.,

$$f \circ \alpha = D\alpha \circ \lambda_{\mu\Sigma} \circ H_\Sigma(f)$$

or, equivalently, for all $c : e \to s \in F$,

$$f \circ c^{\mu\Sigma} = (D\alpha)_s \circ \lambda_{\mu\Sigma,s} \circ \iota_c \circ f_e.$$

There is a unique λ-recursive function $f : \mu\Sigma \to D(\mu\Sigma)$.

Proof. Lemma 23.1 (3). ❏

Theorem 25.4

Let $\Sigma = (S, F)$ be a destructive signature and $\lambda : TH_\Sigma \to H_\Sigma T$ be a distributive law of some endofunctor T on Set^S over H_Σ.

Let $\nu\Sigma$ be final in Alg_Σ. Then $\beta : \nu\Sigma \to H_\Sigma(\nu\Sigma)$ with $H_\Sigma(\nu\Sigma)_s = \prod_{d:s\to e\in F} \nu\Sigma_e$ and $\beta_s = \langle d^{\nu\Sigma}\rangle_{d:s\to e\in F}$ for all $s \in S$ is final in $coAlg_{H_\Sigma}$ (see chapter 15).

An S-sorted function $f : T(\nu\Sigma) \to \nu\Sigma$ is **λ-corecursive** if (f, β) is a λ-bialgebra, i.e.,

$$\beta \circ f = H_\Sigma(f) \circ \lambda_{\nu\Sigma} \circ T\beta$$

or, equivalently, for all $d : s \to e \in F$,

$$d^{\nu\Sigma} \circ f = f_e \circ \pi_d \circ \lambda_{\nu\Sigma,s} \circ (T\beta)_s.$$

There is a unique λ-corecursive function $f : T(\nu\Sigma) \to \nu\Sigma$.

Proof. Lemma 23.1 (4). ❑

Example (sum of streams; see section 20 and [26])

Let X be a semiring. The functions $in : X \to X^{\mathbb{N}}$ and $+ : X^{\mathbb{N}} \times X^{\mathbb{N}} \to X^{\mathbb{N}}$ are defined as follows: Let $\beta = \langle head, tail \rangle : X^{\mathbb{N}} \to H_{Stream(X)}(X^{\mathbb{N}}) = X \times X^{\mathbb{N}}$.

Define $T : Set \to Set$ and $\lambda : TH_{Stream(X)} \to H_{Stream(X)}T$ as follows:

For all $A \in Set$, $h \in Mor(Set)$ and $((x,a),(y,b)) \in (X \times A) \times (X \times A) = TH_{Stream(X)}A$,

$$T(A) = A \times A, \quad T(h) = h \times h,$$
$$\lambda_A((x,a),(y,b)) = (x+y,(a,b)) \in X \times (A \times A) = H_{Stream(X)}TA.$$

A function $+ : X^{\mathbb{N}} \times X^{\mathbb{N}} \to X^{\mathbb{N}}$ satisfies the equations

$$head(s+s') = head(s) + head(s') \quad (1)$$
$$tail(s+s') = tail(s) + tail(s') \quad (2)$$

iff $+$ is a λ-corecursive, i.e., the following equations hold true:

$$\begin{aligned} head \circ + &= \pi_{head} \circ \lambda_{\nu\Sigma} \circ (\beta \times \beta) \\ tail \circ + &= + \circ \pi_{tail} \circ \lambda_{\nu\Sigma} \circ (\beta \times \beta) \end{aligned}$$

Proof. For all $s, s' \in X^{\mathbb{N}}$,

$$\begin{aligned} &(head \circ +)(s, s') = head(s + s'), \\ &(\pi_{head} \circ \lambda_{\nu\Sigma} \circ (\beta \times \beta))(s, s') \\ &\quad = \pi_{head}(\lambda_{\nu\Sigma}((head(s), tail(s)), (head(s'), tail(s')))) \\ &\quad = \pi_{head}(head(s) + head(s'), (tail(s), tail(s'))) = head(s) + head(s'), \\ &(tail \circ +)(s, s') = tail(s + s'), \\ &(+ \circ \pi_{tail} \circ \lambda_{\nu\Sigma} \circ (\beta \times \beta))(s, s') \\ &\quad = +(\pi_{tail}(\lambda_{\nu\Sigma}((head(s), tail(s)), (head(s'), tail(s'))))) \\ &\quad = +(\pi_{tail}(head(s) + head(s'), (tail(s), tail(s')))) = tail(s) + tail(s'). \end{aligned}$$

Hence by Theorem 25.4, equations (1) and (2) have a unique solution $+$.

26.1 Queues

(see [151], p. 185)

A specification of queues with entries from a set A:

```
BEGIN Queue[ A : TYPE ] : CLASSSPEC
  METHOD
    put : [Self, A] -> Self;
    top : Self -> Lift[[A,Self]];

  CONSTRUCTOR
    new : Self;

  ASSERTION SELFVAR x : Self
    q_empty : x.top ≅ ⊥ IMPLIES
      FORALL(a : A) . x.put(a).top ≅ up(a,x);

    q_filled :
      FORALL(a1:A, y : Self) . x.top ≅ up(a1,y) IMPLIES
        FORALL(a2 : A) . x.put(a2).top ≅ up(a1, y.put(a2));

  CREATION
    q_new : new.top ≅ ⊥;
END Queue
```

Signature $Queue = (S, F)$:

$$S = \{queue\}, \quad F = \{ \begin{array}{l} new : queue, \\ top : queue \to 1 + (A \times queue), \\ put : queue \times A \to queue \ \} \end{array}$$

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 [151], p. 181):

$$\begin{array}{lcll} M_{queue} & = & (\mathbb{N} \cup \{\omega\}) \times (\mathbb{N} \to A), & \\ new^M & = & (0, \lambda i.a) & \text{for some } a \in A. \end{array}$$

For all $(n, f) \in M_{queue}$ and $a \in A$,

$$top^M(n, f) = \begin{cases} (\epsilon, 1) & \text{if } n = 0, \\ ((f(0), \lambda i.f(i+1)), 2) & \text{otherwise,} \end{cases}$$

$$put^M((n, f), a) = \begin{cases} (n, f) & \text{if } n = \omega, \\ (n+1, \lambda i.\mathit{if}\ \ i = n\ \mathit{then}\ a\ \mathit{else}\ f(i)) & \text{otherwise.} \end{cases}$$

Hutton's motto:

$$\begin{array}{rcccl} \text{denotational semantics} & = & \text{folding of syntax trees} & = & \text{evaluation in an algebra} \\ \text{operational semantics} & = & \text{unfolding to transition trees} & = & \text{execution in a coalgebra} \end{array}$$

26.2 Arithmetic expressions

([78], sections 2-4)

Let B be a set, $\Sigma = (S, F)$ with

$$S = \{exp\}, \quad F = \{add : exp \times exp \to exp\},$$

$\Sigma(B) = (S, F)$ with

$$S = \{exp\}, \quad F = \{ \begin{array}{l} val : B \to exp, \\ add : exp \times exp \to exp \ \}. \end{array}$$

$\Sigma(B, V) = (S, F)$ with

$$S = \{exp\}, \quad F = \{ \begin{array}{l} val : B \to exp, \\ var : V \to exp, \\ add : exp \times exp \to exp \ \}. \end{array}$$

Hence for all sets and functions X,

$$\begin{aligned} H_\Sigma(X) &= X \times X, \\ H_{\Sigma(B)}(X) &= B + (X \times X), \\ H_{\Sigma(B,V)}(X) &= B + V + (X \times X) \end{aligned}$$

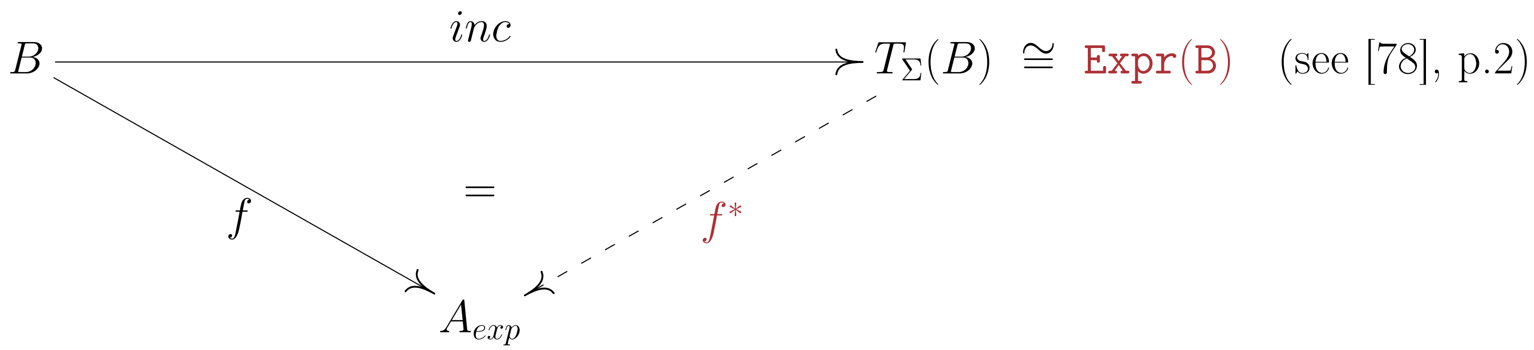
(see chapter 15).

Let A be a $\Sigma(B)$-algebra.

Then $[val^A, add^A] : H_{\Sigma(B)}(A) \to A$ is the corresponding $H_{\Sigma(B)}$-algebra.

Let A be a Σ-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 Σ-algebra over B:



For all $t, t' \in T_\Sigma(B)$, $add^{T_\Sigma(B)}(t,t') = add(t,t')$.

For all $t, t' \in T_\Sigma(B)$ and $b \in B$,

$$f^*(val(b)) = f(b) \text{ and } f^*(add(t,t')) = add^A(f^*(t), f^*(t')).$$

$T_\Sigma(B)$ is also a initial $\Sigma(B)$-algebra:

For all $b \in B$, $val^{T_\Sigma(B)}(b) = val(b)$.

Let A be a $\Sigma(B)$-algebra.

The unique $\Sigma(B)$-homomorphism $fold^A : T_\Sigma(B) \to A$ is defined as follows: For all $b \in B$ and $t, t' \in T_\Sigma(B)$,

$$fold^A(val(b)) = val^A(b) \text{ and } fold^A(add(t,t')) = add^A(fold^A(t), fold^A(t')).$$

$fold^A =$ `deno` (see [78], p.2) for $f = val^A$ and $g = add^A$.

The functor $T_\Sigma =$ `Expr` $: Set \to Alg_\Sigma$ is left adjoint to the forgetful functor from Alg_Σ to Set. For all functions $f : A \to C$, $T_\Sigma(f) = (inc \circ f)^* : T_\Sigma(A) \to T_\Sigma(C)$.

Let A be a $\Sigma(B)$-algebra.

$$
\begin{array}{ccc}
H_{\Sigma(B)}(T_\Sigma(B)) & \xrightarrow{[val^{T_\Sigma(B)},\, add^{T_\Sigma(B)}]} & T_\Sigma(B) \\
{\scriptstyle H_{\Sigma(B)}(fold^A)} \downarrow & & \downarrow {\scriptstyle fold^A} \\
H_{\Sigma(B)}(A) & \xrightarrow{[val^A,\, add^A]} & A
\end{array}
$$

The $\Sigma(\mathbb{Z})$-algebra $\mathbb{Z}$ of integers: $val^{\mathbb{Z}} = id_{\mathbb{Z}},\ add^{\mathbb{Z}} = (+)$

$$fold^{\mathbb{Z}} = id^*_{\mathbb{Z}} = \texttt{eval} : T_\Sigma(\mathbb{Z}) \to \mathbb{Z} \text{ (see [78], p.2)}$$

The $\Sigma(\mathbb{Z}, V)$-algebra $\mathbb{Z}^V$ of integer stores:

$$
\begin{array}{rcl}
val^{\mathbb{Z}^V} & = & \lambda n.\lambda s.n, \\
var^{\mathbb{Z}^V} & = & \lambda x.\lambda s.s(x), \\
add^{\mathbb{Z}^V} & = & \lambda(f,g).\lambda s.f(s) + g(s).
\end{array}
$$

The $\Sigma(\mathbb{Z}, V)$-algebra Com^* of assembler programs:

$$
\begin{array}{rcl}
val^{Com^*} & = & \lambda n.Push(n), \\
var^{Com^*} & = & \lambda x.Load(x), \\
add^{Com^*} & = & \lambda(c,c').c \cdot c' \cdot Add.
\end{array}
$$

Let B be a set, $\Sigma' = (S, F)$ with

$$S = \{state\}, \quad F = \{\delta : state \to state^*\},$$

$\Sigma'(B) = (S, F)$ with

$$S = \{state\}, \quad F = \{ \; \beta : state \to B, \\ \delta : state \to state^* \; \}.$$

Hence the functors $H_{\Sigma'}, H_{\Sigma'(B)} : Set \to Set$ (see chapter 15) are defined as follows:

For all sets and functions X,

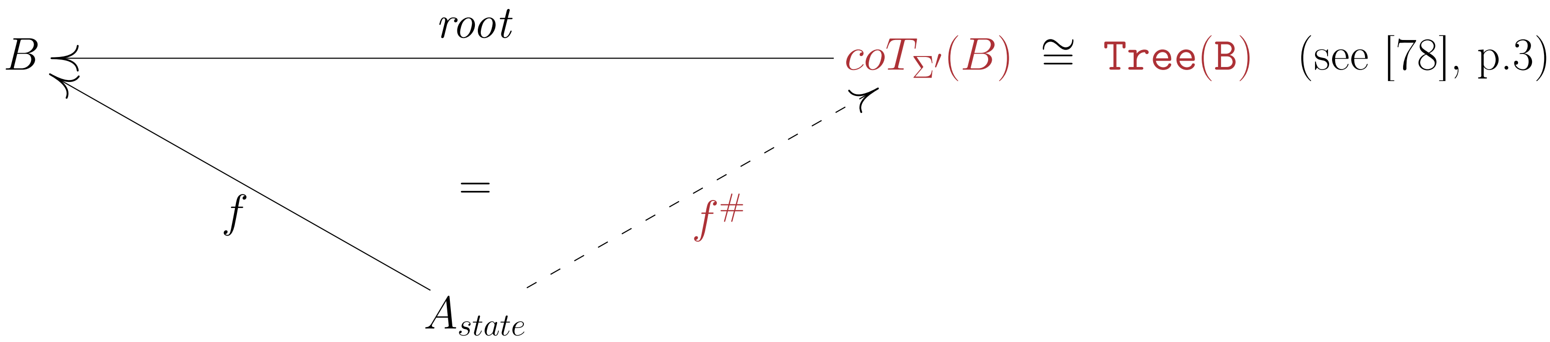
$$\begin{aligned} H_{\Sigma'}(X) &= X^*, \\ H_{\Sigma'(B)}(X) &= B \times X^*. \end{aligned}$$

Let A be a $\Sigma'(B)$-algebra.

Then $\langle \beta^A, \delta^A \rangle : A \to H_{\Sigma'(B)}(A)$ is the corresponding $H_{\Sigma'(B)}$-coalgebra.

Let A be a Σ'-algebra and $coT_{\Sigma'}(B)$ be the set of all finite and infinite trees whose nodes are labelled with B.

$coT_{\Sigma'}(B)$ is the cofree Σ'-algebra over B:



For all $b \in B$, $t_1,\dots,t_n \in coT_{\Sigma'}(B)$, $\delta^{coT_{\Sigma'}(B)}(b(t_1,\dots,t_n)) = (t_1,\dots,t_n)$.

For all $a, a_1,\dots,a_n \in A_{state}$,

$$\delta^A(a) = (a_1,\dots,a_n) \;\Rightarrow\; f^{\#}(a) = f(a)(f^{\#}(a_1),\dots,f^{\#}(a_n)).$$

$coT_{\Sigma'}(B)$ is also a final $\Sigma'(B)$-algebra:

For all $t \in coT_{\Sigma'}(B)$, $\beta^{coT_{\Sigma'}(B)}(t) = root(t)$.

Let A be a $\Sigma'(B)$-algebra.

The unique $\Sigma'(B)$-homomorphism $unfold^A : A \to coT_{\Sigma'}(B)$ is defined as follows: For all $a, a_1,\dots,a_n \in A_{state}$,

$$\delta^A(a) = (a_1,\dots,a_n) \;\Rightarrow\; unfold^A(a) = \beta^A(a)(unfold^A(a_1),\dots,unfold^A(a_n)).$$

$unfold^A =$ `oper` (see [78], p.4) for $f = \beta^A$ and $g = \delta^A$.

The functor $coT_{\Sigma'} = \texttt{Tree} : Set \to Alg_{\Sigma'}$ is right adjoint to the forgetful functor from $Alg_{\Sigma'}$ to Set.

For all functions $f : A \to C$, $coT_{\Sigma'}(f) = (f \circ root)^{\#} : coT_{\Sigma'}(A) \to coT_{\Sigma'}(C)$.

Let A be a $\Sigma'(B)$-algebra.

$$\begin{array}{ccc}
coT_{\Sigma'}(B) & \xrightarrow{\langle \beta^{coT_{\Sigma'}(B)}, \delta^{coT_{\Sigma'}(B)} \rangle} & H_{\Sigma'(B)}(coT_{\Sigma'}(B)) \\
\uparrow{\scriptstyle unfold^A} & & \uparrow{\scriptstyle H_{\Sigma'(B)}(unfold^A)} \\
A & \xrightarrow{\langle \beta^A, \delta^A \rangle} & H_{\Sigma'(B)}(A)
\end{array}$$

The $\Sigma'(T_\Sigma(\mathbb{Z}))$-algebra $T_\Sigma(\mathbb{Z})$:

$$
\begin{array}{rcl}
\beta^{T_\Sigma(\mathbb{Z})} & = & id_{T_\Sigma(\mathbb{Z})} \\
\delta^{T_\Sigma(\mathbb{Z})} : T_\Sigma(\mathbb{Z}) & \rightarrow & T_\Sigma(\mathbb{Z})^* \quad = \\
\texttt{trans} :: \texttt{Expr Int} & \texttt{->} & \texttt{[Expr Int]} \qquad\qquad\qquad \text{(see [78], p.3)} \\
val(n) & \mapsto & \epsilon, \\
add(t,t') & \mapsto & \begin{cases} val(m+n) & \text{if } \exists\, m,n \in \mathbb{Z} : \\ & \quad val(m) = t \wedge val(n) = t', \\ map(\lambda x.add(x,t'))(\delta^{T_\Sigma(\mathbb{Z})}(t)) \cdot & \\ map(\lambda x.add(t,x))(\delta^{T_\Sigma(\mathbb{Z})}(t')) & \text{otherwise} \end{cases}
\end{array}
$$

$$unfold^A = id^{\#}_{T_\Sigma(\mathbb{Z})} = \texttt{exec} : T_\Sigma(\mathbb{Z}) \rightarrow coT_{\Sigma'}(T_\Sigma(\mathbb{Z})) \text{ (see [78], p.4)}$$

The $\Sigma(\mathbb{Z})$-algebra $A = coT_{\Sigma'}(\mathbb{Z})$:

Let $\Psi = \Sigma(\mathbb{Z}) \cup \Sigma'(\mathbb{Z})$. Since $T_\Sigma(\mathbb{Z})$ is initial in $Alg_{\Sigma'(\mathbb{Z})}$ and $\mathcal{A}$ is final in $Alg_{\Sigma'(\mathbb{Z})}$, Theorems 16.1 and 16.3 imply that the conjunction of the following equations has unique solutions both in $T_\Sigma(\mathbb{Z})$ and $\mathcal{A}$:

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, Theorem 16.3 (12) implies

$$unfold^{T_\Sigma(\mathbb{Z})} = fold^A : T_\Sigma(\mathbb{Z}) \to A.$$

More examples

26.3 CCS

(see Calculus of Communicating Systems; [111]; [78], sections 5-8; [126], section 4.4)

$$
\begin{array}{llll}
\mathbf{P} & ::= & \mathbf{N} & -\textit{constants} \\
& | & \alpha.\mathbf{P} & -\textit{prefixing} \\
& | & \sum_{i\in I}\mathbf{P}_i & -\textit{(finite) choice} \\
& | & \mathbf{P}\mid\mathbf{P} & -\textit{parallelism} \\
& | & \mathbf{P}\backslash\alpha & -\textit{restriction} \\
& | & \mathbf{P}[f] & -\textit{relabelling}
\end{array}
$$

$$\frac{P_j \xrightarrow{a} P_j'}{\sum_{i\in I} P_i \xrightarrow{a} P_j'}\ (j\in I)$$

$$\frac{P \xrightarrow{a} P'}{P\mid Q \xrightarrow{a} P'\mid Q} \qquad \frac{Q \xrightarrow{a} Q'}{P\mid Q \xrightarrow{a} P\mid Q'} \qquad \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\bar a} Q'}{P\mid Q \xrightarrow{\tau} P'\mid Q'}$$

$$\frac{P \xrightarrow{b} P'}{P\backslash a \xrightarrow{b} P'\backslash a}\ (a,\bar a \neq b) \qquad \frac{P \xrightarrow{a} P'}{P[f] \xrightarrow{f(a)} P'[f]}$$

FMS version ([125], p. 161)

$$
\begin{array}{rcll}
a(x).P \xrightarrow{a(v)} P[v/x] & & & (\textit{read}) \\
\overline{a}(e).P \xrightarrow{\overline{a}(val(e))} P & & & (\textit{write}) \\
P_1 + P_2 \xrightarrow{a} Q & \Longleftrightarrow & P_1 \xrightarrow{a} Q \ \vee\ P_2 \xrightarrow{a} Q & (\textit{select}) \\
P|P' \xrightarrow{a} Q|P' & \Longleftrightarrow & P \xrightarrow{a} Q & (\textit{parallelize}) \\
P'|P \xrightarrow{a} P'|Q & \Longleftrightarrow & P \xrightarrow{a} Q & \\
P_1|P_2 \xrightarrow{\tau} Q_1|Q_2 & \Longleftrightarrow & P_1 \xrightarrow{a} Q_1 \ \wedge\ P_2 \xrightarrow{\overline{a}} Q_2 & (\textit{communicate}) \\
P \setminus M \xrightarrow{a} Q \setminus M & \Longleftrightarrow & P \xrightarrow{a} Q \ \wedge\ a \in Act \setminus M \setminus \{\overline{b} \mid b \in M\} & (\textit{restrict}) \\
A \xrightarrow{a} Q & \Longleftrightarrow & P \xrightarrow{a} Q \quad \text{if } A \text{ is defined by the equation } A = P & (\textit{call})
\end{array}
$$

$$H_{Proc(Act)} = \lambda A.Act + A^2 + A^2 + A \times Act + A \times Act^{Act}$$

$T_{Proc(Act)}$ is an initial $Proc(Act)$-algebra.

$$H_{Trans(Act)} = \lambda A.(Act \times A)^*$$

The $Trans(Act)$-algebra $DTree(Act)$ is defined as follows:

- $DTree(Act)_{tree}$ is the set of $<$-based labelled trees t over $(\mathbb{N}, Act + 1)$ such that $t(w) = ()$ only if $w = \epsilon$.
- For all $t \in DTree(Act)_{tree}$,

  $$denode^{DTree(Act)}(t) = ((t(i), \lambda w.t(iw)))_{i=1}^{n}$$

  where $n = max(def(t) \cap \mathbb{N})$.

$DTree(Act)$ is final in $Alg_{Trans(Act)}$.

Proof. Let A be a $Trans(Act)$-algebra. A function $unfold^A : A \to DTree(Act)$ is defined as follows:

For all $a \in A_{tree}$, $i > 0$ and $w \in \mathbb{N}^+$, $denode^A(a) = ((x_i, a_i))_{i=1}^n$ implies

$$unfold^A(a)(i) = \begin{cases} x_i & \text{if } 1 \le i \le n, \\ \bot & \text{otherwise,} \end{cases}$$

$$unfold^A(a)(iw) = \begin{cases} unfold^A(a_i)(w) & \text{if } 1 \le i \le n, \\ \bot & \text{otherwise.} \end{cases}$$

Let $a \in A_{tree}$ and

$$denode^A(a) = ((x_i, a_i))_{i=1}^n. \tag{1}$$

$unfold^A$ is $Trans(Act)$-homomorphic: By (1) and the definition of $unfold^A$,

$$max(def(unfold^A(a)) \cap \mathbb{N}) = n.$$

Hence

$$denode^{DTree(Act)}(unfold^A(a)) \overset{\text{Def. } denode^{DTree(Act)}}{=} ((unfold^A(a)(i), \lambda w.unfold^A(a)(iw)))_{i=1}^n,$$

$$\overset{\text{Def. } unfold^A}{=} ((x_i, \lambda w.unfold^A(a_i)(w)))_{i=1}^n = ((x_i, unfold^A(a_i)))_{i=1}^n$$

$$= unfold^A(((x_i, a_i))_{i=1}^n) \overset{(1)}{=} unfold^A(denode^A(a)).$$

unfold^A is unique: Let $h : A \to DTree(Act)$ be a $Trans(Act)$-homomorphism. Then

$$denode^{DTree(Act)}(h(a)) \stackrel{h\ Trans(Act)-hom.}{=} h(denode^A(a)) \stackrel{(1)}{=} h(((x_i, a_i))_{i=1}^n) = ((x_i, h(a_i)))_{i=1}^n. \tag{2}$$

For all $w \in \mathbb{N}^+$,

$$h(a)(w) = unfold^A(a)(w) \tag{3}$$

Proof by induction on $|w|$. For all $1 \leq i \leq n$,

$$h(a)(i) \stackrel{Def.\ denode^{DTree(Act)}}{=} \pi_1(\pi_i(denode^{DTree(Act)}(h(a)))) \stackrel{(2)}{=} \pi_1(\pi_i(((x_i, h(a_i)))_{i=1}^n))$$
$$= \pi_1(x_i, h(a_i)) = x_i \stackrel{Def.\ unfold^A}{=} unfold^A(a)(i).$$

By (2) and the definition of $denode^{DTree(Act)}$,

$$max(def(h(a)) \cap \mathbb{N}) = n. \tag{4}$$

Hence for all $i > n$ and $w \in \mathbb{N}^*$,

$$h(a)(iw) \stackrel{(4)}{=} \bot \stackrel{Def.\ unfold^A}{=} unfold^A(a)(iw).$$

Moreover, for all $1 \le i \le n$ and $w \in \mathbb{N}^*$,

$$h(a)(iw) = (\lambda w.h(a)(iw))(w) \stackrel{Def.\ denode^{DTree(Act)}}{=} \pi_2(\pi_i(denode^{DTree(Act)}(h(a))))(w)$$
$$\stackrel{(2)}{=} \pi_2(\pi_i(((x_i, h(a_i)))_{i=1}^n))(w) = \pi_2(x_i, h(a_i))(w) = h(a_i)(w)$$
$$\stackrel{ind.\ hyp.}{=} unfold^A(a_i)(w) \stackrel{Def.\ unfold^A}{=} unfold^A(a)(iw).$$

Hence (3) holds true. ❑

Trace semantics of processes = final nondeterministic acceptor

Let *Path* $= \bigcup\{def(t) \mid t \in otr(Act \times \mathbb{N}, 1)\}$ (see chapter 3). The $NMed^*(Act)$-algebra *Traces* is defined as follows (see chapter 8):

- $Traces(state) = \mathcal{P}(Path)$.
- For all $W \subseteq Path$ and $x \in Act$,

$$\delta^{Traces}(W)(x) = (\{w \in Path \mid \exists\ i > 0 : (x,i)w \in W\})_{i=1}^n$$

  where $n = max\{i > 0 \mid (x,i)w \in W\}$.

Traces is final in $Alg_{NMed^*(Act)}$.

Proof. Let A be an $NMed^*(Act)$-algebra. A function $unfold^A : A \to Traces$ is defined as follows: For all $a \in A_{state}$,

$$\begin{aligned} unfold^A(a) &= 1 \cup \{(x,i)w \mid x \in Act,\ \delta^A(a)(x) = (a_1,\dots,a_n), \\ &\qquad 1 \le i \le n,\ w \in unfold^A(a_i)\}. \end{aligned}$$

Let $a \in A_{state}$, $x \in Act$ and

$$\delta^A(a)(x) = (a_1,\dots,a_n). \tag{1}$$

$unfold^A$ is $NAcc$-homomorphic: By (1) and the definition of $unfold^A$,

$$max\{i > 0 \mid (x,i) \in unfold^A(a)\} = n.$$

Hence

$$\begin{aligned} &\delta^{Traces}(unfold^A(a))(x) \overset{Def.\ \delta^{Traces}}{=} (\{w \in Path \mid (x,i)w \in unfold^A(a)\})_{i=1}^n \\ &\overset{Def.\ unfold^A}{=} (\{w \in Path \mid w \in unfold^A(a_i)\})_{i=1}^n = (unfold^A_{state}(a_i))_{i=1}^n \\ &= unfold^A_{state^*}(a_1,\dots,a_n) \overset{(1)}{=} unfold^A_{state^*}(\delta^A(a)(x)) = unfold^A_{(state^*)^{Act}}(\delta^A(a))(x). \end{aligned}$$

unfold^A is unique: Let $h : A \to \mathit{Traces}$ be an $\mathit{NMed}^*(\mathit{Act})$-homomorphism. Then

$$\begin{aligned}\delta^{\mathit{Traces}}(h(a))(x) &\overset{h\ \mathit{NMed}^*(\mathit{Act})-hom.}{=} h_{(state^*)^{Act}}(\delta^A(a))(x) = h_{state^*}(\delta^A(a)(x)) \\ &\overset{(1)}{=} h_{state^*}(a_1,\dots,a_n) = (h(a_1),\dots,h(a_n)).\end{aligned} \tag{2}$$

For all $w \in \mathit{Path}$,

$$w \in h(a) \quad \Leftrightarrow \quad w \in \mathit{unfold}^A(a). \tag{3}$$

Proof by induction on $|w|$. By the definition of Traces, (3) holds true for $w = \epsilon$.

By (2) and the definition of δ^{Traces},

$$max\{i > 0 \mid (x,i) \in h(a)\} = n. \tag{4}$$

Let $i > n$ and $v \in \mathit{Path}$. By the definition of unfold^A, $(x,i)v \notin \mathit{unfold}^A(a)$. 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 $v \in \mathit{Path}$,

$$\begin{aligned}(x,i)v \in h(a) &\overset{Def.\ \delta^{\mathit{Traces}}}{\Leftrightarrow} v \in \pi_i(\delta^{\mathit{Traces}}(h(a))(x)) \overset{(2)}{\Leftrightarrow} v \in h(a_i) \\ &\overset{ind.\ hyp.}{\Leftrightarrow} v \in \mathit{unfold}^A(a_i) \overset{Def.\ \mathit{unfold}^A}{\Leftrightarrow} (x,i)v \in \mathit{unfold}^A(a).\end{aligned}$$

Hence (3) holds true for $w = (x,i)v$. ❑

The following Haskell functions `trans'` and `trans` implement the T-coalgebra

$$denode^{T_{Proc(Act)}} : T_{Proc(Act)} \to T(T_{Proc(Act)}) = (Act \times T_{Proc(Act)})^*$$

(see [78], 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

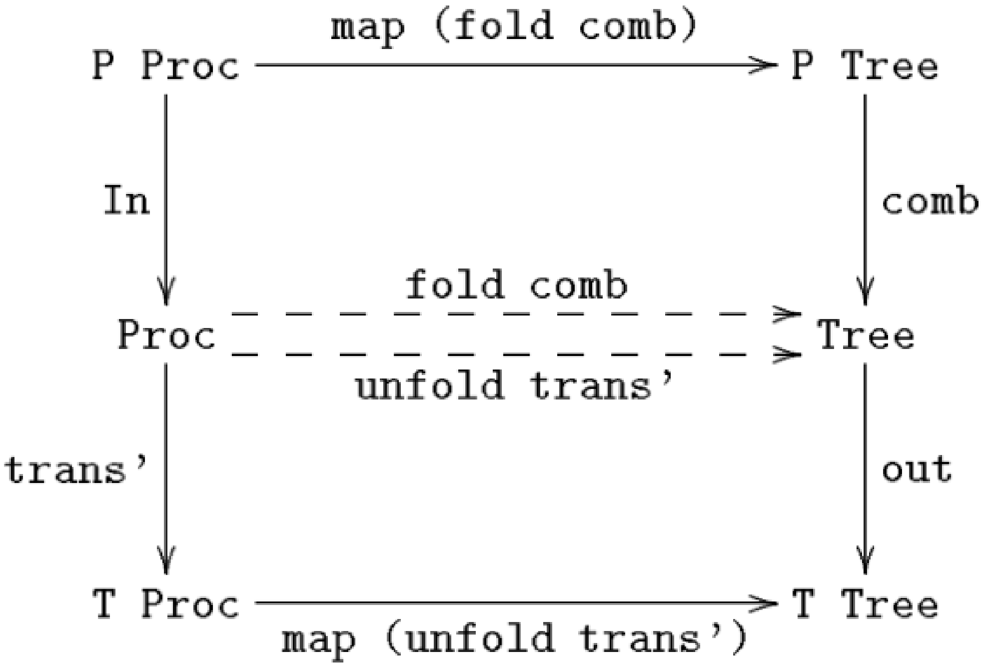
$$[pre^{DTree(Act)}, cho^{DTree(Act)}, par^{DTree(Act)}, res^{DTree(Act)}, rel^{DTree(Act)}] : \\ Act + DTree(Act)^2 + DTree(Act)^2 + DTree(Act) \times Act + DTree(Act) \times Act^{Act} \\ \to DTree(Act)$$

(see [78], 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]))
```

Moreover, Theorem 16.3 (12) implies

$(\texttt{unfold trans}' =)\ unfold^{T_{Proc(Act)}} = fold^{DTree(Act)}\ (= \texttt{fold comb}) : T_{Proc(Act)} \to DTree(Act$



`In` and `out` implement

$$[pre^{T_{Proc(Act)}}, cho^{T_{Proc(Act)}}, par^{T_{Proc(Act)}}, res^{T_{Proc(Act)}}, rel^{T_{Proc(Act)}}] :$$
$$Act + T^2_{Proc(Act)} + T^2_{Proc(Act)} + T_{Proc(Act)} \times Act + T_{Proc(Act)} \times Act^{Act} \to Proc(Act)$$

and $denode^{DTree(Act)} : DTree(Act) \to T(DTree(Act)) = (Act \times DTree(Act))^*$, respectively.

Let $E : V \to T_{Proc(Act)}(V)$ be a system of iterative $Proc(Act)$-equations.

E turns $T_{Proc(Act)}(V)$ into a $Trans(Act)$-algebra: Let F be the set of arrows of $Proc(Act)$.

For all $f : e \to proc \in F$, $t \in T_{Proc(Act)}(V)_e$ and $x \in V_{proc}$,

$$\begin{aligned} denode^{T_{Proc(Act)}(V)}(ft) &= (t, f), \\ denode^{T_{Proc(Act)}(V)}(x) &= denode^{T_{Proc(Act)}(V)}(E(x)). \end{aligned}$$

(1) $V \stackrel{inc_V}{\to} T_{Proc(Act)}(V) \stackrel{unfold^{T_{Proc(Act)}(V)}}{\longrightarrow} DTree(Act)$ solves E in $DTree(Act)$.

Proof. Let $x \in V_{proc}$, $E(x) = ft$ and $t = (t_1, \dots, t_n)$. Hence

$$denode^{T_{Proc(Act)}(V)}(x) = (t, f) \tag{2}$$

and thus for all $i > 0$ and $w \in \mathbb{N}^+_{>0}$,

$$\begin{aligned} &(unfold^{T_{Proc(Act)}(V)} \circ inc_V)^*(E(x))(i) = (unfold^{T_{Proc(Act)}(V)} \circ inc_V)^*(ft)(i) \\ &= f^{DTree(Act)}((unfold^{T_{Proc(Act)}(V)} \circ inc_V)^*(t))(i) \stackrel{Def.\ f^{DTree(Act)}}{=} f \\ &\stackrel{Def.\ unfold^{T_{Proc(Act)}(V)},(2)}{=} unfold^{T_{Proc(Act)}(V)}(x)(\epsilon) \end{aligned}$$

and for all $i \in \mathbb{N}$ and $w \in \mathbb{N}^*$,

$$(unfold^{T_{Proc(Act)}(V)} \circ inc_V)^*(E(x))(iw) = (unfold^{T_{Proc(Act)}(V)} \circ inc_V)^*(ft)(iw)$$
$$= f^{CT_\Sigma}((unfold^{T_{Proc(Act)}(V)} \circ inc_V)^*(u))(iw)$$
$$\stackrel{Def.\ f^{CT_\Sigma}}{=} \left\{ \begin{array}{ll} (unfold^{T_{Proc(Act)}(V)} \circ inc_V)^*(u_i)(w) & \text{if } u = (u_1, \dots, u_n) \text{ and } 1 \leq i \leq n \\ \perp & \text{otherwise} \end{array} \right\}$$
$$\stackrel{Def.\ unfold^{T_{Proc(Act)}(V)},(2)}{=} unfold^{T_{Proc(Act)}(V)}(x)(iw).$$

Therefore, $E_{CT_\Sigma}(unfold^{T_{Proc(Act)}(V)} \circ inc_V) = (unfold^{T_{Proc(Act)}(V)} \circ inc_V)^* \circ E$
$= unfold^{T_{Proc(Act)}(V)} \circ inc_V$, i.e., (1) holds true. ❏

27 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 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] S. Awodey, *Category Theory*, Oxford University Press (2009)

[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. Cancila, 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*, Journal ACM 39 (1992) 95-146

[55] J.A. Goguen, J.W. Thatcher, E.G. Wagner, J.B. Wright, *Initial Algebra Semantics and Continuous Algebras*, Journal ACM 24 (1977) 68-95

[56] R. Goldblatt, *A Calculus of Terms for Coalgebras of Polynomial Functors*, Elsevier ENTCS 44 (2001) 161-184

[57] Andrew D. Gordon, Bisimilarity as a Theory of Functional Programming, Theoretical Computer Science 228 (1999) 5-47

[58] G. Gottlob, C. Koch, R. Pichler, *XPath Processing in a Nutshell*, SIGMOD Record 32 (2003) 21-27

[59] H.P. Gumm, T. Schröder, *Coalgebras of bounded type*, Math. Structures in Computer Science 12 (2002) 565-578

[60] H.P. Gumm, *State Based Systems are Coalgebras*, Cubo - Matematica Educacional 5 (2003) 239-262

[61] H.P. Gumm, *Equational and implicational classes of coalgebras*, Theoretical Computer Science 260 (2001) 57-69

[62] H.P. Gumm, *Universelle Coalgebra*, in: Th. Ihringer, *Allgemeine Algebra*, Heldermann Verlag 2003

[63] G. Gupta et al., *Infinite Computation, Co-induction and Computational Logic*, Proc. CALCO 2011, Springer LNCS 6859 (2011) 40-54

[64] J.V. Guttag, E. Horowitz, and D.R. Musser, *Abstract Data Types and Software Validation*, Communications of the ACM Vol. 21 (1978) 1048-1064

[65] T. Hagino, *Codatatypes in ML*, J. Symbolic Computation 8 (1989) 629-650

[66] H.H. Hansen, C. Kupke, J. Rutten, *Stream Differential Equations: Specification Formats and Solution Methods*, 2016

[67] H.H. Hansen, J. Rutten, *Symbolic Synthesis of Mealy Machines from Arithmetic Bitstream Functions*, Scientific Annals of Computer Science (2010) 97-130

[68] I. Hasuo, B. Jacobs, A. Sokolova, *Generic Trace Theory*, Proc. CMCS 2006, Elsevier ENTCS 164, 47-65

[69] 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

[70] D. Hausmann, T. Mossakowski, L. Schröder, *Iterative Circular Coinduction for CoCasl in Isabelle/HOL*, Proc. FASE 2005, Springer LNCS 3442 (2005) 341-356

[71] 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

[72] R. Hinze, *Adjoint Folds and Unfolds—An extended study*, Science of Computer Programming 78 (2013) 2108-2159

[73] R. Hinze, *Reasoning about codata*, Third Central European Functional Programming School, Springer LNCS 6299 (2010) 42-93

[74] R. Hinze, *Functional Pearl: Streams and Unique Fixed Points*, Proc. 13th ICFP (2008) 189-200

[75] R. Hinze, D.W.H. James, *Proving the Unique-Fixed Point Principle Correct*, Proc. 16th ICFP (2011) 359-371

[76] R. Hinze, N. Wu, J. Gibbons, *Conjugate Hylomorphisms*, Proc. ACM POPL (2015) 527-538

[77] J.E. Hopcroft, R. Motwani, J.D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Prentice Hall 2006

[78] G. Hutton, *Fold and unfold for program semantics*, Proc. 3rd ICFP (1998) 280-288

[79] G. Hutton, *A tutorial on the universality and expressiveness of fold*, J. Functional Programming 9 (1999) 355–372

[80] B. Jacobs, *Invariants, Bisimulations and the Correctness of Coalgebraic Refinements*, Proc. Algebraic Methodology and Software Technology, Springer LNCS 1349 (1997) 276-291

[81] B. Jacobs, *Introduction to Coalgebra*, Cambridge University Press 2017

[82] B. Jacobs, *Exercises in Coalgebraic Specification*, in [?], pp. 237-280

[83] 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

[84] B. Jacobs, *Trace Semantics for Coalgebras*, CMCS 2004

[85] 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

[86] G. Jarzembski, *A new proof of Reiterman's theorem*, Cahiers de topologie et géométrie différentielle catégoriques 35 (1994) 239-247

[87] R. Y. Kain, Automata theory: machines and languages, McGraw-Hill 1972

[88] S. Kamin, *Final data types an their specification extension*, ACM Trans. on Prog. Lang. and Systems Comp. Syst. Sci. 5 (1983) 97-123

[89] S.C. Kleene, *Introduction to Metamathematics*, Van Nostrand 1952

[90] B. Klin, *Structural Operational Semantics for Weighted Transition Systems*, Mosses Festschrift, Springer LNCS 5700 (2009) 121–139

[91] B. Klin, *Bialgebras for structural operational semantics: An introduction*, Theoretical Computer Science 412 (2011) 5043-5069

[92] D. Kozen, *Realization of Coinductive Types*, Proc. Math. Foundations of Prog. Lang. Semantics 27, Carnegie Mellon University, Pittsburgh 2011

[93] C. Kupke, M. Niqui, J. Rutten, *Stream Differential Equations: concrete formats for coinductive definitions*, 2011

[94] C. Kupke, Y. Venema, *Coalgebraic automata theory: basic results*, Logical Methods in Computer Science 4 (2008) 1–43

[95] A. Kurz, *Specifying coalgebras with modal logic*, Theoretical Computer Science 260 (2001) 119–138

[96] A. Kurz, J. Velebil, *Relation lifting, a survey*, Journal of Logical and Algebraic Methods in Programming 85 (2016) 475–499

[97] J. Lambek, *A fixpoint theorem for complete categories*, Math. Zeitschrift 103 (1968) 151-161

[98] J.-L. Lassez, V.L. Nguyen, E.A. Sonenberg, *Fixed Point Theorems and Semantics: A Folk Tale*, Information Processing Letters 14 (1982) 112-116

[99] F.W. Lawvere, *Diagonal arguments in cartesian closed categories*, Reprints in Theory and Applications of Categories 15 (2006) 1–13

[100] D.J. Lehmann, M.B. Smyth, *Algebraic Specification of Data Types: A Synthetic Approach*, Math. Systems Theory 14 (1981) 97-139

[101] S. Mac Lane, *Categories for the Working Mathematician*, Graduate Texts in Mathematics. Springer 1971,1997

[102] Z. Manna, *Mathematical Theory of Computation*, McGraw-Hill 1974

[103] E.G. Manes, M.A. Arbib *Algebraic Approaches to Program Semantics*, Springer 1986

[104] G. Markowsky, *Chain-complete posets and directed sets with applications*, Algebra Universalis 6 (1976) 53-68

[105] M. Marx, M. de Rijke, *Semantic Characterizations of Navigational XPath*, SIGMOD Record 34 (2005) 41-46

[106] B. Mazur, *When is One Thing Equal to Some Other Thing?*, in: B. Gold, R.A. Simons, eds., Proof and other Dilemmas, The Mathematical Association of America 2008

[107] E. Meijer, M. Fokkinga, and R. Paterson, *Functional programming with bananas, lenses, envelopes and barbed wire*, Proc. FPCA 1991, Springer LNCS 523 (1991) 124-144

[108] E. Meijer, G. Hutton, *Bananas in Space: Extending Fold and Unfold to Exponential Types*, Proc. FPCA '95, ACM Publications (1995) 324-333

[109] B. Milewski, *Category Theory for Programmers*, https://bartoszmilewski.com/2014/10/28/category-theory-for-programmers-the-preface

[110] D.A. Miller, G. Nadathur, *Higher-order logic programming*, Proc. ICALP 1986, Springer LNCS (1986) 448-462

[111] R. Milner, *Communication and Concurrency*, Prentice-Hall 1989

[112] E. Moggi, *Notions of computation and monads*, Information and Computation 93 (1991) 55-92

[113] F.L. Morris, *Advice on Structuring Compilers and Proving Them Correct*, Proc. ACM POPL (1973) 144-152

[114] T. Mossakowski, L. Schröder, M. Roggenbach, H. Reichel, *Algebraic-coalgebraic specification in CoCASL*, J. Logic and Algebraic Programming 67 (2005) 146-197

[115] G. Nadathur, D. Miller, *Higher-Order Logic Programming*, D.M. Gabbay, C.J. Hogger, eds., Handbook of Logic in Artificial Intelligence and Logic Programming 5, Clarendon Press (1998) 499-590

[116] D. Orchard, *Should I use a Monad or a Comonad?*, submitted to MSFP 2012

[117] P. Padawitz, *Church-Rosser-Eigenschaften von Graphgrammatiken und Anwendungen auf die Semantik von LISP*, Diplomarbeit, TU Berlin 1978

[118] P. Padawitz, Computing in Horn Clause Theories, EATCS Monographs on Theoretical Computer Science 16, Springer-Verlag, 1988 (free copies are available from the author)

[119] P. Padawitz, Deduction and Declarative Programming, Cambridge Tracts in Theoretical Computer Science 28, Cambridge University Press, 1992

[120] P. Padawitz, *Inductive Theorem Proving for Design Specifications*, J. Symbolic Computation 21 (1996) 41-99

[121] P. Padawitz, *Proof in Flat Specifications*, in: [18], pp. 321-384

[122] P. Padawitz, Swinging Types = Functions + Relations + Transition Systems, Theoretical Computer Science 243 (2000) 93-165

[123] P. Padawitz, *Expander2: program verification between interaction and automation*, slides for [131], WFLP 2006

[124] P. Padawitz, *Expander2: Two inductive proofs*, Video, TU Dortmund 2017

[125] P. Padawitz, *Formale Methoden des Systementwurfs*, TU Dortmund 2007

[126] P. Padawitz, *Swinging Types At Work*, TU Dortmund 2008

[127] P. Padawitz, *Swinging Data Types*, TU Dortmund 2009

[128] P. Padawitz, *Dialgebraic Specification and Modeling*, TU Dortmund 2010

[129] 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)

[130] P. Padawitz, *From grammars and automata to algebras and coalgebras*, Proc. CAI 2011, Springer LNCS 6742 (2011) 21-43

[131] P. Padawitz, *Expander2 as a Prover and Rewriter*, TU Dortmund 2012

[132] P. Padawitz, *From fixpoint to predicate co/induction and its use in standard models*, TU Dortmund 2014

[133] P. Padawitz, *(Co)Algebraic Specification with Base Sets, Recursive and Iterative Equations*, IFIP WG 1.3 Meeting 2014

[134] P. Padawitz, *Modeling and reasoning with I-polynomial data types*, IFIP WG 1.3 Meeting + CMS 2016

[135] P. Padawitz, *Modellieren und Implementieren in Haskell*, Technical Report No. 868, TU Dortmund

[136] P. Padawitz, *Übersetzerbau (Algebraic Compiler Construction*, TU Dortmund 2016

[137] P. Padawitz, *Logik für Informatiker (Logic for Computer Scientists)*, Technical Report No. 867, TU Dortmund

[138] P. Padawitz, *From Modal Logic to (Co-)Algebraic Reasoning*, TU Dortmund 2017

[139] 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

[140] D. Pattinson, L. Schröder, *Program Equivalence is Coinductive*, Proc. 21st LICS (2016), 337-346

[141] D. Pavlovic and M. Escardó, *Calculus in coinductive form*, Proc. 13th LICS (1998), 408-417

[142] 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)

[143] S. Phillips, W.H. Wilson, *Categorial Compositionality: A Category Theory Explanation for the Systematicity of Human Cognition*, PLoS Computational Biology 6 (2010)

[144] B. Pierce, *Basic Category Theory for Computer Scientists*, MIT Press 1991

[145] G.D. Plotkin, J. Power, *Tensors of comodels and models for operational semantics*, Elsevier ENTCS 218 (2008) 295–311

[146] C. Pulte, *Natürliche Transformationen*, Lecture in the Proseminar *Kategorientheoretische Grundlagen*, TU Dortmund 2012

[147] H. Reichel, *An Approach to Object Semantics based on Terminal Coalgebras*, Mathematic Structures in Computer Science 5 (1995) 129-152

[148] H. Reichel, *Dialgebraic Logics*, Elsevier ENTCS 11 (1998) 1-9

[149] H. Reichel, *An Algebraic Approach to Regular Sets*, in: K. Futatsugi et al., Goguen Festschrift, Springer LNCS 4060 (2006) 449-458

[150] J. Reiterman, *The Birkhoff theorem for finite algebras*, Algebra Universalis 14 (1982) 1-10

[151] J. Rothe, H. Tews, B. Jacobs, *The Coalgebraic Class Specification Language CCSL*, Journal of Universal Computer Science 7 (2001) 175-193

[152] W.C. Rounds, *Mappings and Grammars on Trees*, Mathematical Systems Theory 4 (1970) 256-287

[153] J. Rutten, *Processes as terms: non-wellfounded models for bisimulation*, Math. Struct. in Comp. Science 15 (1992) 257-275

[154] J. Rutten, *Automata and Coinduction (an exercise in coalgebra)*, Proc. CONCUR '98, Springer LNCS 1466 (1998) 194–218

[155] J. Rutten, *Automata, Power Series, and Coinduction: Taking Input Derivatives Seriously*, Proc. ICALP '99, Springer LNCS 1644 (1998) 645-654

[156] J. Rutten, *Universal coalgebra: a theory of systems*, Theoretical Computer Science 249 (2000) 3-80

[157] J. Rutten, *Behavioral differential equations: a coinductive calculus of streams, automata, and power series*, Theoretical Computer Science 308 (2003) 1-53

[158] J. Rutten, *On Streams and Coinduction*, in: *Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems*, CRM Monograph Series 23 (2004) 1-92

[159] J. Rutten, *A coinductive calculus of streams*, Math. Struct. in Comp. Science 15 (2005) 93-147

[160] J. Rutten, *The Method of Coalgebra: exercises in coinduction*, CWI Amsterdam (2019)

[161] J. Salamanca, M. Bonsangue, J. Rutten, *Equations and Coequations for Weighted Automata*, Proc. MFCS 2015, Springer LNCS 9234 (2015) 444–456

[162] D. Sannella, A. Tarlecki, *Foundations of Algebraic Specification and Formal Software Development*, Springer 2012

[163] D. Schwencke, *Coequational logic for accessible functors*, Information and Computation 208 (2010) 1469–1489

[164] T. Schwentick, *XPath query containment*, SIGMOD Record 33 (2004) 101–109

[165] K. Sen, G. Rosu, *Generating Optimal Monitors for Extended Regular Expressions*, Proc. Runtime Verification 2003, Elsevier ENTCS 89 (2003) 226-245

[166] L. Simon, *Coinductive Logic Programming*, Ph.D. thesis, University of Texas at Dallas (2006)

[167] A. Silva, J. Rutten, *A coinductive calculus of binary trees*, Information and Computation 208 (2010) 578–593

[168] A. Silva, F. Bonchi, M. Bonsangue, J. Rutten, *Quantitative Kleene coalgebras*, Information and Computation 209 (2011) 822-849

[169] 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

[170] D.I. Spivak, T. Giesa, E. Wood, M.J. Buehler, *Category Theoretic Analysis of Hierarchical Protein Materials and Social Networks*, www.plosone.org 2011

[171] D.I. Spivak, R.E. Kent, *Ologs: A Categorical Framework for Knowledge Representation*, www.plosone.org 2012

[172] D.I. Spivak, Category Theory for the Sciences, MIT Press 2014

[173] Th. Streicher, Introduntion to Category Theory and Categorical Logic, University of Darmstadt 2003

[174] A. Tarski, *A lattice-theoretical fixpoint theorem and its applications*, Pacific J. Math. 5 (1955), 285-309

[175] W. Thomas, *Languages, Automata, and Logic*, in: Handbook of Formal Languages, Vol. 3: Beyond Words, Springer (1997) 389-456

[176] W. Thomas, *Applied Automata Theory*, Course Notes, RWTH Aachen (2005)

[177] D. Turi, G. Plotkin, *Towards a Mathematical Operational Semantics*, Proc. 12th LICS (1997) 280-291

[178] 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

[179] 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

[180] T. Uustalu, V. Vene, *Primitive (Co)Recursion and Course-of-Value (Co)Iteration*, INFORMATICA 10 (1999) 5-26

[181] Ph. Wadler, *Theorems for free!*, Proc. FPLCA '89, ACM Press (1989) 347-359

[182] E.G. Wagner, J.B. Wright, J.A. Goguen, J.W. Thatcher, *Some Fundamentals of Order-Algebraic Semantics*, IBM Research Report 6020 (1976)

[183] E.G. Wagner, J.W. Thatcher, J.B. Wright, *Free continuous theories*, IBM Research Report 6906 (1977)

[184] 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

[185] E.G. Wagner, *Algebraic semantics*, in: Handbook of Logic in Computer Science 3: Semantic Structures, Clarendon Press (1994) 323-393

[186] M. Wand, *Final algebra semantics and data type extension*, J. Comp. Syst. Sci. 19 (1979) 27-44

[187] R.F.C. Walters, *Categories and Computer Science*, Cambridge University Press 1992

[188] J. Winter, *Coalgebraic Characterizations of Automata-Theoretic Classes*, Ph.D. thesis, Radboud University Nijmegen 2014

[189] J. Winter, M.M. Bonsague, J. Rutten, *Context-Free Languages, Coalgebraically*, Proc. CALCO 2011

[190] M. Wirsing, *Structured Algebraic Specifications: A Kernel Language*, ?Theoretical Computer Science 42 (1986) 123-249

[191] M. Wirsing, *Algebraic Specification*, in: J. van Leeuwen, ed., Handbook of Theoretical Computer Science, Elsevier (1990) 675-788

[192] N.S. Yanofsky, *A Universal Approach to Self-Referential Paradoxes, Incompleteness and Fixed Points*, The Bulletin of Symbolic Logic 9 (2003) 362-386