プログラマのための余代数

山崎利治

今日の話題

代数と余代数はともに集合と関数の組として定義される. 簡単に言えば, それらはつぎである.

$$\text{代数} \qquad (X, FX \xrightarrow{a} X)$$
$$\text{余代数} \qquad (X, X \xrightarrow{c} FX)$$

ここで X は集合, F は集合上の構成子あるいは分解子, a, c は関数である. プログラマになじみの

抽象データ型は代数

状態系は余代数

である. また, 自己参照を伴う無限の要素をもつデータ型やオブジェクト指向のオブジェクトも余代数とみなされる. 今日の話題はこの余代数である.

今日の話題

余代数に関連するいくつかの概念があり, この説明が目的である.

	代数	余代数
構造	$a : FX \to X$	$c : X \to FX$
意味	始代数	終余代数
定義・証明原理	合同関係	双模倣関係
	帰納法	余帰納法

話の予定

I　余代数の例

II　グラフ, 圏, 関手

III　余代数

IV　双模倣関係と余帰納法

I 余代数の例

状態系

余代数は状態系の理論である

状態系とは

- 系は一定の操作をもち
- 系の操作を通して環境と相互作用を行い
- 見えない内部状態に依存した挙動を示し
- 一般に停止しない

ものである．その例をつぎに挙げる．

暗箱

暗箱に二つの釦 h, t と表示窓がある．釦 h を押すと文字を窓に表示する．釦 t を押すと状態が変わる．これはつぎのように表せる:

$$\begin{cases} h & : S \to A \\ t & : S \to S \end{cases}$$

纏めてつぎのように表せる.

$$\boxed{(S, \langle h, t\rangle : S \to A \times S)}$$

ここで, S は状態集合, A は表示文字集合である.

暗箱

たとえば

$$\begin{tikzcd} 1 \arrow[r, "a"] & 2 \arrow[r, bend left, "b"] & 3 \arrow[l, bend left, "c"] & 4 \arrow[l, "b"'] & 5 \arrow[l, "c"'] & 6 \arrow[l, "a"'] \end{tikzcd}$$

$$\left(s \xrightarrow{a} s' \Leftrightarrow h(s) = a \wedge t(s) = s'\right)$$

上の図で, 状態 1 と 6 は h 釦を押しただけでは a を窓に観察して区別できないが t, h と釦を押すと一方は b, もう一方は c を観察し区別できる. 状態 2 と 4 は以降の挙動が全く同じで区別できない. 状態間の区別できないという関係 (挙動同値) を $\sim$ と記せば

$$s \sim s' \Leftrightarrow h(s) = h(s') \wedge t(s) \sim t(s')$$

である.

有限オートマトン

有限オートマトンは順序回路や語句解析の思考道具として利用されてきた.

I 上の**決定性有限オートマトン** $A = (I, S, s_0, F, \delta)$

I : 入力文字集合

S : 状態集合

s_0 : 始状態 $\in S$

F : 終状態集合 $\subseteq S$

$\delta : S \times I \to S$ 状態推移関数

有限オートマトン

$\delta : S \times I \to S$ をつぎのように拡張する.

$$\delta^* : S \times I^* \to S \qquad (I^* \text{ は } I \text{ の文字の有限列集合})$$

ただし, δ^* はつぎを満たす.

$$\delta^*(s, 1) = s \qquad (1 \text{ は空列})$$

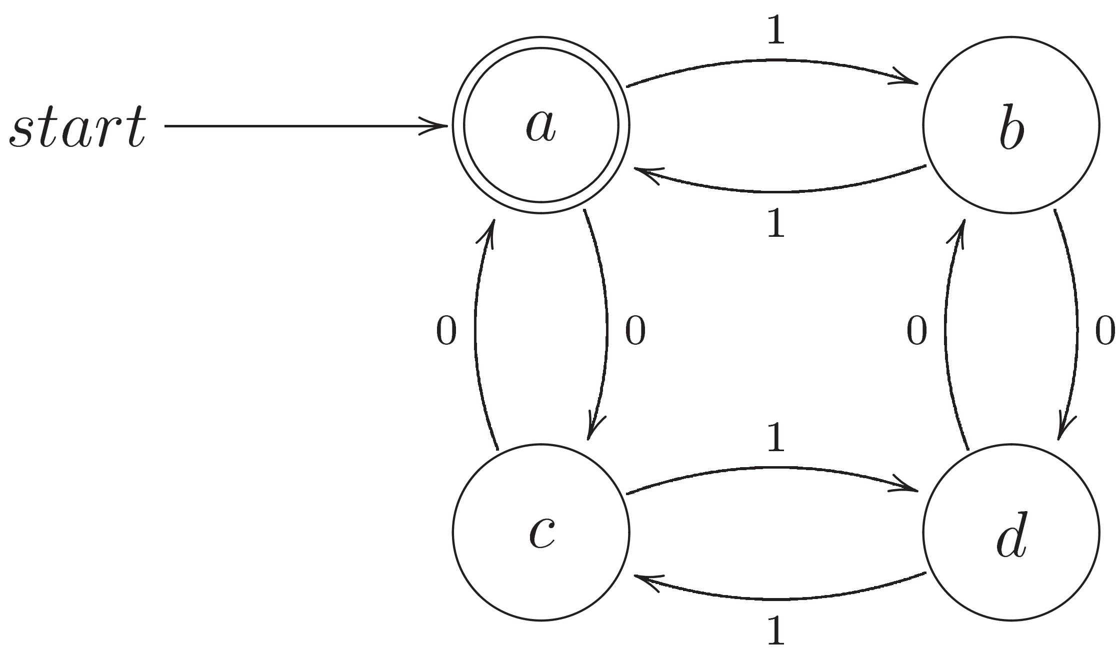
$$\delta^*(s, i) = \delta(s, i) \qquad (i \in I)$$

$$\delta^*(\delta^*(s, \alpha), \beta) = \delta^*(s, \alpha\beta) \qquad (\alpha, \beta \in I^*)$$

A の**認識言語** $L(A) \triangleq \{\alpha \in I^* | \delta^*(s_0, \alpha) \in F\}$

有限オートマトン

空列および出現文字をそれぞれ偶数個含む 0,1 有限列を認識する決定性有限オートマトン (二重丸は終状態)



$$[\ (1(00)^* + 0(11)^*10(00)^*)(01(11)^*10(00)^*)^*1\ +\ 0(11)^*0$$
$$+\ (1(00)^* + 0(11)^*10(00)^*)(01(11)^*10(00)^*)^*01(11)^*0]^*$$

有限オートマトン

決定性有限オートマトンの関数 $\delta : S \times I \to S$ を
$\delta' : S \to S^I \;\; (\delta'(s) \triangleq \delta(s, -) : I \to S)$ と考える.
決定性有限オートマトンはつぎのように表せる.

$$\boxed{(S, \alpha : S \to S^I \times \mathbb{B})}$$

$\mathbb{B} = \{true, false\}$ は終状態を判定するために用いた
$(\langle \delta', final \rangle : S \to S^I \times \mathbb{B})$.

札付推移系

札付推移系は並行プロセスの模型として利用されている.

札付推移系は $(S, L, T, s_0 \in S)$ である. ここで

S : 状態集合

L : 札集合

T : 推移関係 $\subseteq S \times L \times S$

s_0 : 始状態 $\in S$

$(s, l, s') \in T$ を $s \xrightarrow{l} s'$ と記す.

札付推移系

例 $S = \{s_0, s_1, s_2, s_3\}, L = \{coin, paleale, bitterale, relish, change\}$

$T =$

$\{(s_0, coin, s_1), (s_1, paleale, s_2), (s_1, bitterale, s_2), (s_2, relish, s_3), (s_3, change, s_0)\}$

$$\begin{tikzcd}
start \arrow[r] & s_0 \arrow[r, "coin"] & s_1 \arrow[d, bend right, "paleale"'] \arrow[d, bend left, "bitterale"] \\
 & s_3 \arrow[u, "change"] & s_2 \arrow[l, "relish"]
\end{tikzcd}$$

札付推移系

札付推移系は纏めると

$$\boxed{(S, \gamma : S \to \mathbb{P}(L \times S))}$$

関係 T を $S \to \mathbb{P}(L \times S)$ ($\mathbb{P}X$ は X の巾集合) と関数に変更した.

Kripke 枠と Kripke 模型

Kripke 枠は様相論理の模型のための構造である.

Kripke 枠

(W, R) ここで

W : 可能世界集合

R : 可達関係 $\subseteq W \times W$

Kripke 模型

(W, R, V) ここで

A : 素命題集合 $\quad \ni \; a, b, \ldots$

V : 付値 $A \to \mathbb{P}W$ ($\mathbb{P}W$ は W の巾集合)

Kripke 枠と Kripke 模型

様相論理　命題論理につぎを追加

$\Box p \cdots$ 必然的に p である

$\Diamond p \cdots$ p である可能性がある

構文

a を素命題,p, q を様相論理式 (その全体を F) とする.

$$p, q \ ::= \ a \mid p \wedge q \mid \neg p \mid \Box p$$

意味

与えられた Kripke 模型 (W, R, V) に対して世界と様相論理式の間の関係 $\models$ をつぎのように定める.($w \models p \cdots$ p は w で妥当,w は p を充足する)

$p, q \in F, w \in W$ に対して

$$w \models a \Leftrightarrow w \in V(a)$$

$$w \models p \wedge q \Leftrightarrow w \models p \text{ かつ } w \models q$$

$$w \models \neg p \Leftrightarrow w \models p \text{ ではない}$$

$$w \models \Box p \Leftrightarrow wRw' \text{ となるすべての } w' \text{ に対して } w' \models p$$

Kripke 枠と Kripke 模型

たとえば,

$W \triangleq \{a, b, c\}$

$$R \triangleq \begin{tikzcd} a \arrow[loop above] \arrow[r] & b \arrow[loop above] \arrow[r] & c \arrow[loop above] \end{tikzcd}$$

1) (W, R) **で** $\Box p \Rightarrow p$ **は恒真**

2) (W, R) **で** $\Box p \Rightarrow \Box\Box p$ **は偽**

2) $\models$ **を**

$$a \models p, \quad b \models p, \quad c \not\models p$$

とすれば,aRy **となるのは** $y = a \vee y = b$**. いずれの場合にも** $y \models p$ **ゆえに** $a \models \Box p$**. かたや** $bRc \wedge c \not\models p$ **ゆえに** $b \models \Box p$**. また** aRb **だから** $a \not\models \Box\Box p$**. したがって** $a \not\models \Box p \Rightarrow \Box\Box p$**.**

Kripke 枠と Kripke 模型

Kripke 模型は纏めると

$$\boxed{(W, \gamma : W \to \mathbb{P}(A \times W))}$$

これは札付推移系に他ならない.

OOP のオブジェクト

つぎの Java のクラス宣言も余代数とみなせる.

```
//class Bank_Account declaration
class BAcnt{
 private int amt;  //field-declaration
 BAcnt(){ amt=0; }  // constructor-declaration
 //method-declarations
 public void deposit(int n){ amt += n; }
 public show(){ return amt }
}
```

OOP のオブジェクト

- `amt` は非公開
- 仕様としてはつぎをいいたい:

`x.deposit(m).deposit(n).show()==x.deposit(m+n).show()`

つぎはいいたくない:

`x.deposit(m).deposit(n)==x.deposit(m+n)`

オブジェクトは纏めると

$$\boxed{(S, \langle show(), deposit(n)\rangle : S \to \mathbb{Z} \times S^{\mathbb{Z}})}$$

無限列

Haskell など遅延評価を行う関数型プログラムでは流 (stream) という
ある型のデータの無限列が扱える.

```
data IS = Cons Int IS
h :: IS -> Int
h (Cons n s) = n
t :: IS -> IS
t (Cons n s) = s
seq :: Int -> Int ->IS
seq n m = Cons n (seq (n+m) m)
-- seq 0 1 == 0,1,2,...
```

無限列

このデータ型無限列 IS はつぎのように表せる.

$$\boxed{IS = (IS, \langle h, t\rangle : IS \to \mathbb{Z} \times IS)}$$

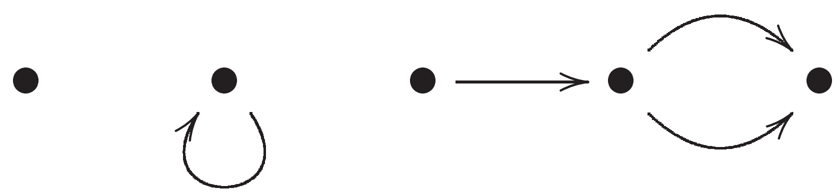
II　グラフ, 圏, 関手

グラフ

グラフ (有向多重グラフ) (E, V, s, t)

E : **辺,** V **点,** $s : E \to V, t : E \to V$ **辺の始点, 終点を示す関数**

例　有限グラフ (辺も点も有限個) は $\bullet$ と $\to$ とによって図示できる.



例　集合上の関係

$A \triangleq \{1, 2, 3\}, B \triangleq \{2, 3, 4\}, R \triangleq \{(1, 2), (2, 2), (2, 3), (1, 4)\}$

$$\begin{tikzcd}
R & 1 \arrow[r] \arrow[d] & 2 \arrow[r] \arrow[loop below] & 3 \\
 & 4 & &
\end{tikzcd}$$

グラフ射

G_1, G_2 : グラフグラフ射 $g : G_1 \to G_2$ とはつぎのような射
$g = (g_E, g_V)$ のこと

$$
\begin{tikzcd}
E_{G_1} \arrow[r, "s_{G_1}"] \arrow[d, "g_E"'] & V_{G_1} \arrow[d, "g_V"] \\
E_{G_2} \arrow[r, "s_{G_2}"'] & V_{G_2}
\end{tikzcd}
\qquad
\begin{tikzcd}
E_{G_1} \arrow[r, "t_{G_1}"] \arrow[d, "g_E"'] & V_{G_1} \arrow[d, "g_V"] \\
E_{G_2} \arrow[r, "t_{G_2}"'] & V_{G_2}
\end{tikzcd}
$$

例　任意のグラフ G からグラフ $H : \begin{tikzcd} v \arrow[loop above, "e"] \end{tikzcd}$ へのグラフ射 g が存在する
(G の点を v に, 辺を e に対応させればよい).

圏

グラフ $\mathbb{C}$ がつぎをみたすとき**圏**という. この場合点を**対象**, 辺を**射**, 辺の始点を**域**, 終点を**余域**という.

- 各対象 A にはそれを域, 余域とする射 (恒等射) 1_A がある.
- $A \xrightarrow{f} B \xrightarrow{g} C$ の状況の二つの射 f, g に対してその合成射 $g \circ f : A \to C$ が存在する.
- $A \xrightarrow{f} B$ のとき $f \circ 1_A = f = 1_B \circ f$
- 合成射についてつぎが成立する. $h \circ (g \circ f) = (h \circ g) \circ f$

圏 $\mathbb{C}$ の対象全体を $Ob(\mathbb{C})$, その射全体を $Mor(\mathbb{C})$ と記し, また域, 余域を A, B とする射の全体を $\mathbb{C}(A, B)$ と記す. 考える圏は小さい ($Ob(\mathbb{C})$ も $Mor(\mathbb{C})$ も集合) とする.

圏

例 Mon 単一対象と任意の数の射がつくる圏 (単位半群)

Set, Set_f 集合を対象, それらの上の関数を射とする圏, 後者は有限集合

Pnf 集合を対象, それらの上の部分関数を射とする圏

Has Haskell の型を対象, 関数を射とする圏

Grf グラフを対象, グラフ射を射とする圏

圏

圏 $\mathbb{C}$ の余圏 $\mathbb{C}^{op}$

1. $Ob(\mathbb{C}^{op}) = Ob(\mathbb{C})$
2. $\mathbb{C}^{op}(B, A) = \mathbb{C}(A, B) \qquad (*)$

$(*)$ は $f : A \to B \ (f \in \mathbb{C}(A, B))$ に対して
$f^{op} : B \to A \ (f^{op} \in C^{op}(B, A))$ が１対１に対応し，かつ，

$$\begin{tikzcd} A \arrow[r, shift left, "f"] & B \arrow[l, shift left, "f^{op}"] \arrow[r, shift left, "g"] & C \arrow[l, shift left, "g^{op}"] \end{tikzcd}$$

$$(g \circ f)^{op} = f^{op} \circ g^{op}$$

を意味する．またとくに

$$(1_A)^{op} = 1_A$$

圏

圏 $\mathbb{C}$ における双対性

$\mathbb{C}$ で成立する性質 P に対してそこに現れる射の向きを逆にしてえられる性質を P^{op} を P の**双対性質**という. 一般の圏においてつぎの**双対原理**が成立つ.

> 圏 $\mathbb{C}$ において, ある命題 P が成立てば,
> その双対命題 P^{op} も成立つ.

始対象と終対象

圏 $\mathbb{C}$ の対象 I が始対象 $\rightleftharpoons$

$$\forall A \in \mathbb{C}\ \exists!\, g_A : I \to A$$

つまり, $\mathbb{C}(I, A) = \{g_A\}$

圏 $\mathbb{C}$ の対象 T が終対象 $\rightleftharpoons$

$$\forall A \in \mathbb{C}\ \exists!\, f_A : A \to T$$

つまり, $\mathbb{C}(A, T) = \{f_A\}$

Set では空集合が始対象である.

Set では単集合 $\{*\}$ が終対象である.

$\exists!\, f$ は「f が一意的に存在する」と読む.

関手

$\mathbb{C}, \mathbb{D}$ を圏とする.$\mathbb{C}$ から $\mathbb{D}$ への **(共変) 関手** $F : \mathbb{C} \to \mathbb{D}$ とはつぎを満たすグラフ射のことである.

- $F(g \circ f) = F(g) \circ F(g)$
- $F(1_A) = 1_{F(A)}$

関手

例

恒等関手 $Id : \mathbb{C} \to \mathbb{C}$

$$f : C \to C' \mapsto f : C \to C'$$

定数関手 $K_D : \mathbb{C} \to \mathbb{D}$

$$f : C \to C' \mapsto 1_D : D \to D$$

Haskell の型構成子 **e.g. [](List)**

関手

例

(共変) 巾関手 $\mathbb{P}(\_) : Set \to Set$

$$S \xrightarrow{f} S' \mapsto \mathbb{P}S \xrightarrow{\mathbb{P}f} \mathbb{P}S' : S \supseteq T \mapsto f(T) \subseteq S'$$

上はつぎを満たす. $\mathbb{P}1_S = 1_{\mathbb{P}S}, \quad \mathbb{P}(g \circ f) = \mathbb{P}g \circ \mathbb{P}f$

(共変) 有限巾関手 $\mathbb{P}_f(\_) : Set \to Set$

合成関手 $\mathbb{C} \xrightarrow{F} D \xrightarrow{G} E$ **に対して** $\mathbb{C} \xrightarrow{G \circ F} E : \; G \circ F(C) \triangleq G(F(C))$

積関手 $F, G : \mathbb{C} \to \mathbb{D}$ **に対して** $F \times G : \mathbb{C} \to \mathbb{D}$ **をつぎのように定義する (**$\mathbb{D}$ **は積をもつとする).**

$$C \xrightarrow{f} C' \mapsto FC \times GC \xrightarrow{Ff \times Gf} FC' \times GC'$$

余積関手も同様に定義する.

関手

多項式関手　自己関手で恒等関手, 定数関手, 積関手, 余積関手, それらの合成による関手 ($(\_)^A : \mathbb{C} \to \mathbb{C} : X \mapsto X^A$ (A を集合として) を含めることがある).

III 余代数

代数と余代数

以下 $\mathbb{C}$ は圏,$F : \mathbb{C} \to \mathbb{C}$ は関手,$A, A', C, C' \in \mathbb{C}, \alpha, \gamma \in Mor\mathbb{C}$ とする.

$(A, \alpha : FA \to A) : F$ 代数
$f : (A, \alpha) \to (A', \alpha') : F$ 代数射はつぎを可換にする射:

$$\begin{tikzcd} FA \arrow[r, "Ff"] \arrow[d, "\alpha"'] & FA' \arrow[d, "\alpha'"] \\ A \arrow[r, "f"'] & A' \end{tikzcd}$$

代数と代数射とは圏 $Alg(\mathbb{C}, F)$ をつくる

$(C, \gamma : C \to FC) : F$ 余代数
$g : (C, \gamma) \to (C', \gamma') : F$ 余代数射はつぎを可換にする射:

$$\begin{tikzcd} C \arrow[r, "g"] \arrow[d, "\gamma"'] & C' \arrow[d, "\gamma'"] \\ FC \arrow[r, "Fg"'] & FC' \end{tikzcd}$$

余代数と余代数射とは圏 $Coalg(\mathbb{C}, F)$ をつくる.

$\mathbb{C}, F$ が明白であるときはそれを省略する.

代数と余代数

注意

- 恒等射は余代数射
- 余代数射の合成は余代数射
- 考える圏は多くは Set または Set_f
- 考える関手は多くは多項式関手か $\mathbb{P}_f$

余代数

データ型 $\mathtt{IS}$ は Set 上の自己関手 $F(\_) = K_{\mathbb{N}} \times Id(\_)$ に対する余代数である．

終余代数

ある圏 $\mathbb{C}$, その上の自己関手 F に対する圏 $Coalg(\mathbb{C}, F)$ には終対象, すなわち, 終余代数が存在する.

★ 多項式関手 F に対する $Coalg(Set, F)$ には終余代数が存在する.

★ 巾関手に対する $Coalg(Set, \mathbb{P})$ には終余代数が存在しない.

★ 有限巾関手に対する $Coalg(Set, \mathbb{P}_{fin})$ には終余代数が存在する.

終余代数

★ 終余代数はすべて同型である.

★ 終余代数の構造射は全単射 (同型余代数射) である.

$$\begin{tikzcd} C \arrow[r, "\gamma"] \arrow[d, "\gamma"'] & FC \arrow[r, "!"] \arrow[d, "F\gamma"] & C \arrow[d, "\gamma"] \\ FC \arrow[r, "F\gamma"'] & FFC \arrow[r, "F!"'] & FC \end{tikzcd}$$

$$! : (FC, F\gamma) \to (C, \gamma) \qquad ! \circ \gamma = 1_C \qquad \gamma \circ ! = 1_{FC}$$

射の分解

Set では写像 $f : A \to B$ がつぎのように分解できる (m, e はそれぞれ単射, 全射).

$$A \rightarrowtail^{m} C \xrightarrow{e} B$$

部分対象と商対象

$A, B \in \mathbb{C}$ 　　e, m **はそれぞれ全射, 単射**

(A, m) **が** B **の部分対象** $\rightleftharpoons$

$$A \overset{m}{\rightarrowtail} B$$

(e, B) **が** A **の商対象** $\rightleftharpoons$

$$A \overset{e}{\twoheadrightarrow} B$$

部分余代数と商余代数

Set の場合, 余代数 (B,β) で単射 $\alpha : A \to B$ は余代数射になる．すなわち,(A,α) は部分余代数になる.

Set の場合, 余代数 (A,α) 上の合同関係 $\sim$ に対して, 標準射 $a : A \to A/\sim$ が余代数射になる．また, つぎを可換にする射 α^* が一意に存在し, すなわち,$(A/\sim, \alpha^*)$ は商余代数になる．

$$
\begin{tikzcd}
A \arrow[r, "a"] \arrow[d, "\alpha"'] & A/\sim \arrow[d, "\exists!\ \alpha^*"] \\
FA \arrow[r, "Fa"'] & F(A/\sim)
\end{tikzcd}
$$

IV 双模倣関係と余帰納法

双模倣関係

起源

- 様相論理 (van Benthem)
- プロセス代数 (Milner,Park)
- 非正則集合 (Aczel)

双模倣関係

動機

プロセスの同値性に関する議論である.プロセスを札付推移系 $(S, \xrightarrow{l} \subseteq S \times L \times S)$ と見做し,プロセス S の状態 s と T の状態 t についてつぎのような関係 $R \subseteq S \times T$ を双模倣的といった.

1. sRt
2. $\forall l \forall s' \bullet s \xrightarrow{l} s' \Rightarrow \exists t' \bullet t \xrightarrow{l} t'$
3. $\forall l \forall t' \bullet t \xrightarrow{l} t' \Rightarrow \exists s' \bullet s \xrightarrow{l} s'$

$$
\begin{tikzcd}
s \arrow[r, "R"] \arrow[d, "l"'] & t \arrow[d, "l"] \\
\forall s' \arrow[r, "R"] & \exists t'
\end{tikzcd}
\qquad
\begin{tikzcd}
s \arrow[r, "R"] \arrow[d, "l"'] & t \arrow[d, "l"] \\
\exists s' \arrow[r, "R"] & \forall t'
\end{tikzcd}
$$

挙動同値と双模倣関係

C, D を F 余代数とする. $c \in C, d \in D$ に対して
$c \sim d$(挙動同値) とは,F 余代数 E と余代数射
$f : C \to E, g : D \to E$ が存在して $f(c) = g(d)$ となること.

同上の状況において
$B \subseteq C \times D$ が双模倣関係とは,F 余代数 $\beta : B \to FB$ が存在して射
影 $\pi_1 : B \to C, \pi_2 : B \to D$ が余代数射になること.

$$\begin{tikzcd}
C \arrow[d, "\gamma"'] & B \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[d, "\beta"] & D \arrow[d, "\delta"] \\
FC & FB \arrow[l, "F\pi_1"] \arrow[r, "F\pi_2"'] & FD
\end{tikzcd}$$

挙動同値と双模倣関係

★ 余代数 (C, γ) 上の対角関係 $\Delta_C \triangleq \{(c, c) | c \in C\}$ は双模倣関係.

★ 双模倣関係の逆関係は双模倣関係.

★ 双模倣関係の合併は双模倣関係.

★ 双模倣関係の合成は双模倣関係.

★ 最大双模倣関係が存在し, それを双模倣性という.

双模倣関係

その例 札付推移系 S, T

$$
\begin{tikzcd}
S & s_0 \arrow[r, "b"] \arrow[d, "a"'] & s_1 \arrow[r, "b"] \arrow[d, "a"'] & \cdots & T & t \arrow[loop above, "b"] \arrow[d, "a"'] \\
 & s'_0 & s'_1 & & & t'
\end{tikzcd}
$$

$R \subseteq S \times S \quad R = \{(s_i, s_j) | i, j \geq 0\} \cup \{(s'_i, s'_j) | i, j \geq 0\}$

$R' \subseteq S \times T \quad R' = \{(s_i, t) | i \geq 0\} \cup \{(s'_i, t') | i \geq 0\}$

$f : S \to T : \ f(s_i) = t, \ f(s'_i) = t'$ **余代数射**

単純, 外延的

F 余代数 C が単純とは, 任意の $c, c' \in C$ に対して $c \sim c'$ ならば $c = c'$ が成立するときにいう.

F 余代数 C が外延的とは, 任意の F 余代数 D に対して $\forall d \in D \exists c \in C \bullet c \sim d$ が成立するときにいう.

単純とは余帰納法証明原理を満たすこと, つまり, $c = c'$ を証明するためには $c \sim c'$ を示せば十分であることである.
外延的とは F に関するすべての可能な挙動を惹き起こすことである.

単純, 外延的

★ 挙動同値関係による商余代数は単純である.

★ 任意の余代数からの余代数射をもつ余代数は外延的である.

終余代数

$\boxed{F\ \text{余代数}\ (C,\gamma)\ \text{が単純かつ外延的} \iff (C,\gamma)\ \text{は終余代数}}$

終余代数 (C,γ) の構造射 $\gamma : C \to FC$ は全単 (同型) 射である (Lambek).

終余代数

★ $Coalg(Set, K_A \times Id)$ においてつぎは終余代数.

$$(A^\omega, \langle h \times t\rangle : A^\omega \to A \times A^\omega) \text{ (}A^\omega \text{ は } A \text{ 上の無限列)}$$

$\because$　任意の F 余代数 $(S, \langle hd, tl\rangle : S \to A \times S)$ に対して $(A^\omega, \langle h \times t\rangle : A^\omega \to A \times A^\omega)$ への余代数射 ϕ が一意に存在する.

$$\begin{tikzcd}
S \arrow[r, "\phi"] \arrow[d, "\langle hd, tl\rangle"'] & A^\omega \arrow[d, "\langle h, t\rangle"] \\
A \times S \arrow[r, "(K_A \times Id)\phi"'] & A \times A^\omega
\end{tikzcd}$$

$(K_A \times Id)\phi = 1_A \times \phi$ だから,$\forall s \in S \quad h(\phi(s)) = hd(s) \quad t(\phi(s)) = \phi(tl(s))$ をいえば ϕ は余代数射になる．また帰納法によって, つぎが示せる.

$$\forall k \in \mathbb{N} \bullet t^k(\phi(a)) = \phi(tl^k(s)) \quad h(\phi(tl^k(s))) = h(t^k(s))$$

これは余代数射の存在と一意性を示す.

終余代数

★ $Coalg(Set, Id \times K_A \times Id)$ においてつぎは終余代数.

A の要素を節の札とする無限木

★ $Coalg(Set, Id^{K_A} \times K_{\mathbb{B}})$ においてつぎは終余代数.

A 上の言語の余代数 $(\mathbb{P}A^*, \gamma : \mathbb{P}A^* \to (\mathbb{P}A)^A \times \mathbb{B})$

ただし,

$$\pi_1 \circ \gamma(L) = Le = \{w \in A^* | ew \in L\}$$

$$\pi_2 \circ \gamma(L) = true \text{ if } e \in L$$

問題

```
zip :: IS -> IS -> IS
zip s1 s2 = Cons (h s1) (zip s2 (t s1))
nat = seq 0 1
even = seq 0 2
odd = seq 1 2
```

`nat == zip even odd` を証明せよ

問題　(余帰納法による証明)

`IS` 上の双模倣は２項関係 $R \subseteq IS \times IS$ でつぎを満たすものである.

$$sRs' \Rightarrow h(s) = h(s') \wedge t(s)\, R\, t(s')$$

余帰納法による証明のために `nat` と `zip even odd` の間の双模倣を見出す必要がある. つぎがそうである.

$$R \triangleq \{(seq\ k\ 1, zip\ (seq\ k\ 2)\ (seq\ (k+1)\ 2)) | k \in \mathbb{N}\}$$

$k \in n$ として, まずつぎがわかる.

$$h(seq\ k\ 1) = k = h(zip\ (seq\ k\ 2)\ (seq\ (k+1)\ 2))$$

問題　(余帰納法による証明)

つぎに

$$t(seq\ k\ 1)\ R\ t(zip\ (seq\ k\ 2)\ (seq\ (k+1)\ 2))$$

はつぎからわかる.

$$\begin{aligned}
&t(seq\ k\ 1) = seq\ (k+1)\ 1 \\
&t(zip\ (seq\ k\ 2)\ (seq\ (k+1)\ 2)) \\
&\qquad = zip\ (seq\ (k+1)\ 2)\ (t(seq\ k\ 2)) \\
&\qquad = zip\ (seq\ (k+1)\ 2)\ ((seq\ (k+2)\ 2))
\end{aligned}$$

問題　(帰納法による証明)

自然数集合 $\mathbb{N}$ 上の帰納法によってつぎの二つの補題を証明する.

- $\forall n \in \mathbb{N} \bullet t^n(nat) = seq\ n\ 1$
- $\forall n \in \mathbb{N} \bullet t^n(zip\ even\ odd)$
  $= zip\ (seq\ n\ 2)(seq\ (n+1)\ 2)$

以上からつぎがえられる.

- $\forall n \in \mathbb{N} \bullet h(t^n(nat) = h(t^n\ zip\ even\ odd))$

つぎを認めれば証明ができる.

$$\textbf{ext} \quad [\forall n \in \mathbb{N} \bullet h(t^n(s)) = h(t^n(s'))] \Rightarrow s = s'$$

参考文献

★ R.Backhouse, et al.(eds.), Algebraic and coalgebraic methods in the mathematics of program construction. LNCS 2297, Springer, 2002.

★ J.Barwise and L.Moss, Vicious circle. CSLI, 1996.

★ R.Milner, Communication and concurrency. Prentice Hall, 1989.