代数・モナド・プログラム

眞田 嵩大
数理解析研究所

学生談話会，2021 年 7 月 8 日

Contents

話の流れ・前提知識

だいたい以下の流れで話します．時間の都合で一部省略するかもしれません．

- 代数に共通する構造を観察．普遍代数を導入する．
- モナドを導入．代数との対応を観察
- プログラムの解釈に代数を利用
- ハンドラが自由代数の普遍性による射とみる
- モナドを拡張して，拡張版の「代数・モナド・プログラム」を紹介

学部程度の数学の知識を仮定して話をします．具体的には

- 群・環を触ったことがある．
- 圏，関手，自然変換，随伴の定義を知っている．

ぐらいを仮定しています．
講演中の質問も歓迎します．

代数

モナド

プログラム

より詳細な構造へ

いろいろな代数構造

モノイド

モノイドとは $(A, e : 1 \to A, m : A \times A \to A)$ **であって**

1. $m(e, a) = a = m(a, e)$ for all $a \in A$
2. $m(a, m(b, c)) = m(m(a, b), c)$ for all $a, b, c \in A$

を満たすもののこと.

群

群とは $(A, e : 1 \to A, m : A \times A \to A, i : A \to A)$ **であって**

1. $m(e, a) = a = m(a, e)$ for all $a \in A$
2. $m(a, m(b, c)) = m(m(a, b), c)$ for all $a, b, c \in A$
3. $m(a, i(a))) = e = m(i(a), a)$ for all $a \in A$

を満たすもののこと.

こういう構造を統一的に扱うことを考える.

シグネチャと項

シグネチャ

シグネチャとは，集合 $\Sigma = \{\sigma_i\}_{i \in I}$ であり，各 σ_i には自然数 $\mathrm{arity}(\sigma_i) \in \mathbb{N}$ が割り当てられているもの．

各 $\sigma_i \in \Sigma$ は “演算の記号” であり，$\mathrm{arity}(\sigma_i)$ はその記号の引数の個数を表している．

項の集合 $\mathrm{Term}_\Sigma(X)$

シグネチャ Σ と集合 X に対して項の集合 $\mathrm{Term}_\Sigma(X)$ を次で定める．

$$\frac{x \in X}{x \in \mathrm{Term}_\Sigma(X)} \qquad \frac{\sigma \in \Sigma \quad \{t_i\}_{i=1}^{\mathrm{arity}(\sigma)} \subset \mathrm{Term}_\Sigma(X)}{\sigma(t_1, \ldots, t_{\mathrm{arity}(\sigma)}) \in \mathrm{Term}_\Sigma(X)}$$

項のイメージ

$\Sigma = \{\sigma, \ldots\}$, $\mathrm{arity}(\sigma) = 2$, $X = \{x, y, z, \ldots\}$ **とする．**

$$\sigma(\sigma(x, y), z) = \left(\begin{array}{ccccc} & & \sigma & & \\ & \sigma & & & z \\ x & & y & & \end{array} \right) \in \mathrm{Term}_{\Sigma}(X)$$

項は，葉が X，中間ノードが Σ でラベル付けられた木である．

等式

等式

$l, r \in \mathrm{Term}_\Sigma(X)$ **の組** (l, r) **を等式と呼び** $X \vdash l = r$ **と書く．**

等式の例

$\Sigma = \{e, m\},\ \mathrm{arity}(e) = 0,\ \mathrm{arity}(m) = 2$ **とする．**

$$\{x\} \vdash m(x, e) = x$$

$$\{x, y\} \vdash m(x, y) = m(y, x)$$

$$\{x, y, z\} \vdash m(m(x, y), z) = m(x, m(y, z))$$

は等式である．

等式の集合 $\{X_i \vdash l_i = r_i\}_i$ **のことを** $\mathcal{E}$ **などと書く．これは代数が満たすべき等式の集合である．**

等式理論

等式理論

等式理論とは，シグネチャと等式の組 $\mathcal{T} = (\Sigma_{\mathcal{T}}, \mathcal{E}_{\mathcal{T}})$ である．

モノイドの等式理論

$$\Sigma_{\mathcal{M}} = \{e, m\},\ \mathrm{arity}(e) = 0,\ \mathrm{arity}(m) = 2$$

$$\mathcal{E}_{\mathcal{M}} = \left\{ \begin{array}{c} \{x\} \vdash m(e, x) = x \\ \{x\} \vdash m(x, e) = x \\ \{x, y, z\} \vdash m(m(x, y), z) = m(x, m(y, z)) \end{array} \right\}$$

とする．このとき等式理論 $\mathcal{M} = (\Sigma_{\mathcal{M}}, \mathcal{E}_{\mathcal{M}})$ が作れる．これをモノイドの等式理論と呼ぶ．

以降，等式理論のことを単に理論とよぶこともある

シグネチャの解釈

以降 $\mathcal{C}$ を積を持つ圏とする（よくわからなければ $\mathcal{C} = \mathbf{Set}$ と考えて大丈夫です）.

解釈

シグネチャ Σ の $\mathcal{C}$ における解釈 A は以下のデータからなる：

- 台対象 $A \in \mathrm{Ob}\,\mathcal{C}$
- 各演算 $\sigma \in \Sigma$ に対して $\mathcal{C}$ の射 $|\sigma|^A : A^{\mathrm{arity}(\sigma)} \to A$

Σ の解釈 A が与えられたとき，$t \in \mathrm{Term}_\Sigma(X)$ に対して $\mathcal{C}$ の射 $|t|^A \colon A^X \to A$ を次のように定義できる：

$$|x|^A = \left(A^X \xrightarrow{\pi_x} A \right)$$

$$|\sigma(t_1, \ldots, t_n)|^A = \left(A^X \xrightarrow{\langle |t_i| \rangle_i} A^{\mathrm{arity}(\sigma)} \xrightarrow{|\sigma|} A \right)$$

解釈の例

モノイドのシグネチャの解釈

モノイドのシグネチャ $\Sigma_{\mathcal{M}} = \{e, m\}$ の Set における解釈 M は以下のデータからなる：

- **集合 M**
- **写像 $|e|^M : \{\star\} \to M$**
- **写像 $|m|^M : M \times M \to M$**

例えば $m(x, e) \in \mathrm{Term}_{\Sigma_{\mathcal{M}}}(\{x\})$ に対して

$$|m(x, e)|^M : M^{\{x\}} \to M$$

は，x への M の元の代入の仕方を決めれば M の元がひとつ定まるということを言っている．
この M が本当にモノイドになるには，上のデータがモノイドの公理を満たすという条件が必要．

理論のモデル

モデル

理論 $\mathcal{T} = (\Sigma, \mathcal{E})$ の圏 $\mathcal{C}$ におけるモデル（または代数）とは，

- **Σ の解釈 A であって，**
- **任意の等式 $(X \vdash l = r) \in \mathcal{E}$ について**

$$|l|^A = |r|^A \quad : A^X \to A$$

が成立するもの．

モデル（代数）を，その等式理論 $\mathcal{T}$ を明示して $\mathcal{T}$ モデル（$\mathcal{T}$ 代数）と書くときもある．

モデルの例

モノイドの理論のモデル

モノイドの理論 $\mathcal{M} = (\{e, m\}, \mathcal{E}_{\mathcal{M}})$ の Set におけるモデル M は，

- **集合 M，写像 $|e|^M \colon \{\star\} \to M$，$|m|^M \colon M \times M \to M$**

であって，以下を満たすもの．つまり通常のモノイドである．

$$|m(e, x)|^M = |x|^M \colon M^{\{x\}} \to M$$
$$|m(x, e)|^M = |x|^M \colon M^{\{x\}} \to M$$
$$|m(m(x, y), z)|^M = |m(x, m(y, z))|^M \colon M^{\{x,y,z\}} \to M$$

例えば一番上の条件は以下と同値：

$$\forall a \in M. |m|^M(|e|^M, a) = a$$

これは $|e|^M$ が左単位元であるという条件である．

モデルの間の射

準同型

A, B を理論 $\mathcal{T}$ の $\mathcal{C}$ におけるモデルとする．A から B への準同型 $f : A \to B$ とは，

- $\mathcal{C}$ における射 $f : A \to B$ であって，
- 各 $\sigma \in \Sigma_{\mathcal{T}}$ について以下の図式が可換であるもの．

$$\begin{array}{ccc} A^{\mathrm{arity}(\sigma)} & \xrightarrow{f^{\mathrm{arity}(\sigma)}} & B^{\mathrm{arity}(\sigma)} \\ \downarrow{\scriptstyle |\sigma|^A} & & \downarrow{\scriptstyle |\sigma|^B} \\ A & \xrightarrow{\quad f \quad} & B \end{array}$$

上の可換図式は Set では

$$f(|\sigma|^A(a_1, \ldots, a_n)) = |\sigma|^B(f(a_1), \ldots, f(a_n))$$

と同じこと．

モデルのなす圏

モデルの圏

$\mathcal{T} = (\Sigma, \mathcal{E})$ を理論とする．理論 $\mathcal{T}$ の圏 $\mathcal{C}$ におけるモデルとその間の準同型たちは圏 $\mathbf{Mod}_{\mathcal{T}}(\mathcal{C})$ をなす．

モノイドの圏

モノイドの理論 $\mathcal{M}$ の圏 Set におけるモデルとその間の準同型の圏 $\mathbf{Mod}_{\mathcal{M}}(\mathbf{Set})$ は，通常のモノイドとその間のモノイド準同型のなす圏である．

他の例：群

群

- シグネチャ $\Sigma_{\mathcal{G}} = \{e, m, i\}$
- 等式 $\mathcal{E}_{\mathcal{G}}$ は

$$\left\{ \begin{array}{c} \{x\} \vdash m(e, x) = x \\ \{x\} \vdash m(x, e) = x \\ \{x, y, z\} \vdash m(m(x, y), z) = m(x, m(y, z)) \\ \{x\} \vdash m(x, i(x)) = e \\ \{x\} \vdash m(i(x), x) = e \end{array} \right\}$$

- 理論 $\mathcal{G} = (\Sigma_{\mathcal{G}}, \mathcal{E}_{\mathcal{G}})$ を群の理論と呼ぶ.
- 理論 $\mathcal{G}$ の Set におけるモデル $G \in \mathbf{Mod}_{\mathcal{G}}(\mathbf{Set})$ は通常の群である.

練習問題：環の理論はどのようなものか？

忘却関手

忘却関手

忘却関手 $U\colon\ \mathbf{Mod}_{\mathcal{T}}(\mathcal{C}) \to \mathcal{C}$ **が** $UA = A$ **として定義できる．**

モノイドの忘却

忘却関手

$$U\colon\ \mathbf{Mod}_{\mathcal{M}}(\mathbf{Set}) \to \mathbf{Set}$$

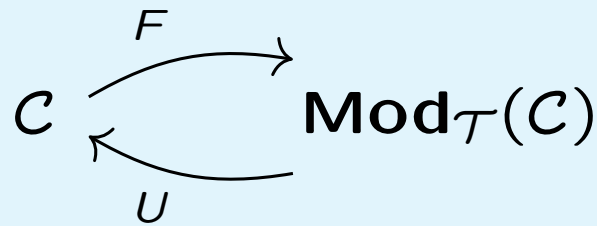
は，通常のモノイドの圏から集合の圏への通常の忘却関手である．

$$UM = M \text{ の台集合}$$

自由 ⊣ 忘却

自由関手

忘却関手 U の左随伴 $F \dashv U$ が存在するとき，それを自由関手と呼ぶ．



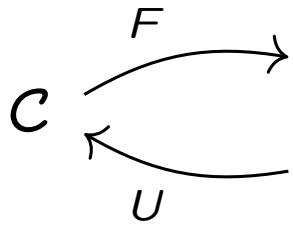
$$\mathbf{Mod}_{\mathcal{T}}(\mathcal{C})(FA, B) \cong \mathcal{C}(A, UB)$$

$\mathcal{C} = \mathbf{Set}$ のとき，自由関手は存在する．

$$F : \mathbf{Set} \to \mathbf{Mod}_{\mathcal{T}}(\mathbf{Set})$$
$$FX = \mathrm{Term}_{\Sigma_{\mathcal{T}}}(X)/\mathcal{E}_{\mathcal{T}}$$

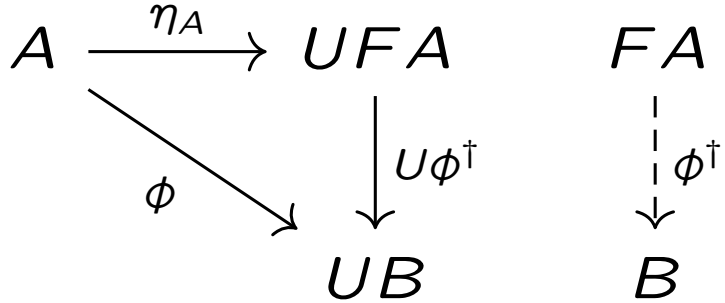
自由代数の普遍性

随伴 $F \dashv U$ は自由生成された代数の普遍性を主張している．



$$\mathbf{Mod}_T(\mathcal{C}) \qquad \mathbf{Mod}_T(\mathcal{C})(FA, B) \cong \mathcal{C}(A, UB)$$

つまり圏 $\mathcal{C}$ の射 $\phi\colon A \to UB$ があれば $\mathbf{Mod}_T(\mathcal{C})$ の射 $f^\dagger\colon FA \to B$ が一意的に存在し，以下の可換図式を満たす．



もっと代数の言葉でいうと，「生成元の行き先さえ定めれば，そこから自由生成された代数の射が一意的に定まる」ということ．

モノイドの自由生成

自由生成されたモノイド

モノイドの理論に対する自由関手

$$F\colon \mathbf{Set} \to \mathbf{Mod}_{\mathcal{M}}(\mathbf{Set})$$

について，FX は X で自由に生成されたモノイドのことである．随伴 $F \dashv U$ から出てくる同型

$$\mathbf{Mod}_{\mathcal{T}}(\mathbf{Set})(FX, M) \cong \mathbf{Set}(X, UM)$$

は，自由モノイドからのモノイド準同型は生成元の行き先を決めれば決まるということを言っている．

代数

モナド

プログラム

より詳細な構造へ

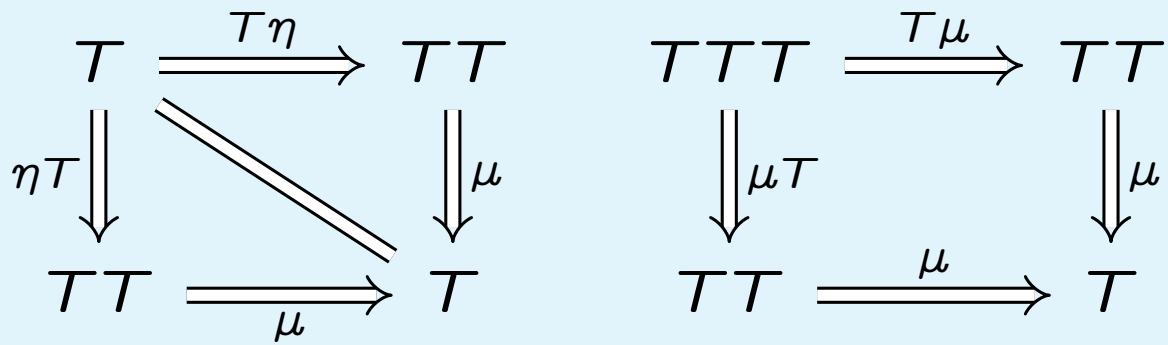
モナドの話

モナド

圏 $\mathcal{C}$ 上のモナドとは関手 $T\colon \mathcal{C} \to \mathcal{C}$ であって，

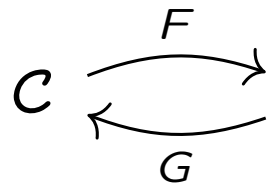
- **自然変換 $\eta\colon \mathrm{Id}_{\mathcal{C}} \Rightarrow T$ （単位）**
- **自然変換 $\mu\colon T \circ T \Rightarrow T$ （掛け算）**

を持ち，以下の可換図式を満たす：



随伴があればモナドができる

随伴 $F \dashv G$ があるとする．このとき関手 $T = G \circ F$ はモナドとなる．



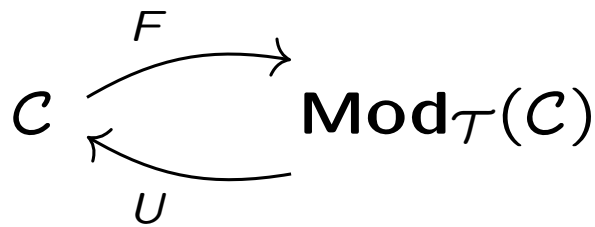
$$\mathcal{D}(FA, B) \cong \mathcal{C}(A, GB)$$

- **単位 $\eta : \mathrm{Id}_{\mathcal{C}} \Rightarrow T$ は** $\underset{\mathrm{id}_{FA}}{\mathcal{D}(FA, FA)} \cong \underset{\eta_A}{\mathcal{C}(A, GFA)}$ **で定義．**
- **掛け算 $\mu : TT \Rightarrow T$ は** $\underset{\varepsilon_B}{\mathcal{D}(FGB, B)} \cong \underset{\mathrm{id}_{GB}}{\mathcal{C}(GB, GB)}$ **を使って $G\varepsilon F : GFGF \Rightarrow GF$ として定義．**

練習問題：この η と μ がモナドの条件を満たすことを確認せよ．

代数の理論からモナドを作る

1. 代数の理論 $\mathcal{T}$ が与えられたとする.
2. $\mathcal{C}$ 上の $\mathcal{T}$ のモデルのなす圏 $\mathbf{Mod}_{\mathcal{T}}(\mathcal{C})$ と忘却関手 $U\colon\ \mathbf{Mod}_{\mathcal{T}}(\mathcal{C}) \to \mathcal{C}$ ができる.
3. U の左随伴 F が存在するとする（特に $\mathcal{C} = \mathbf{Set}$ のときは常に存在する）.

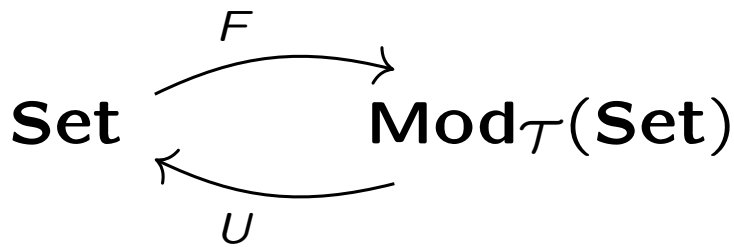


4. $\mathcal{C}$ 上のモナド $T = U \circ F$ ができる.

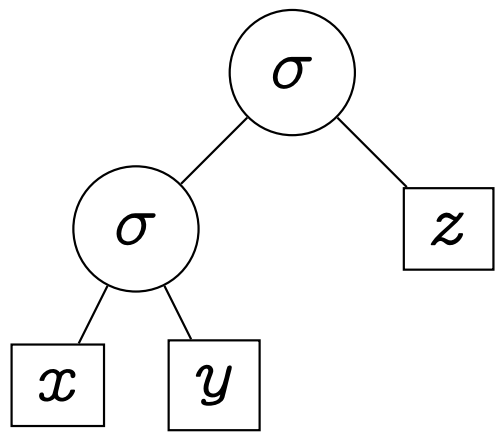
$\mathcal{C} = \mathbf{Set}$ として T がどのようなものか観察しよう.

理論からできたモナドの観察

モナド $T = U \circ F$ は，集合 $X \in \mathbf{Set}$ を $TX = U(FX) = U(\mathrm{Term}_{\Sigma_T}(X)/\mathcal{E}_T)$ に送る．これは X から自由生成してできた代数の台集合をとる関手である．



簡単のため $\mathcal{E}_T = \varnothing$ とする．$\mathrm{Term}_{\Sigma_T}(X)$ の元は木とみることができるのであった．



単位

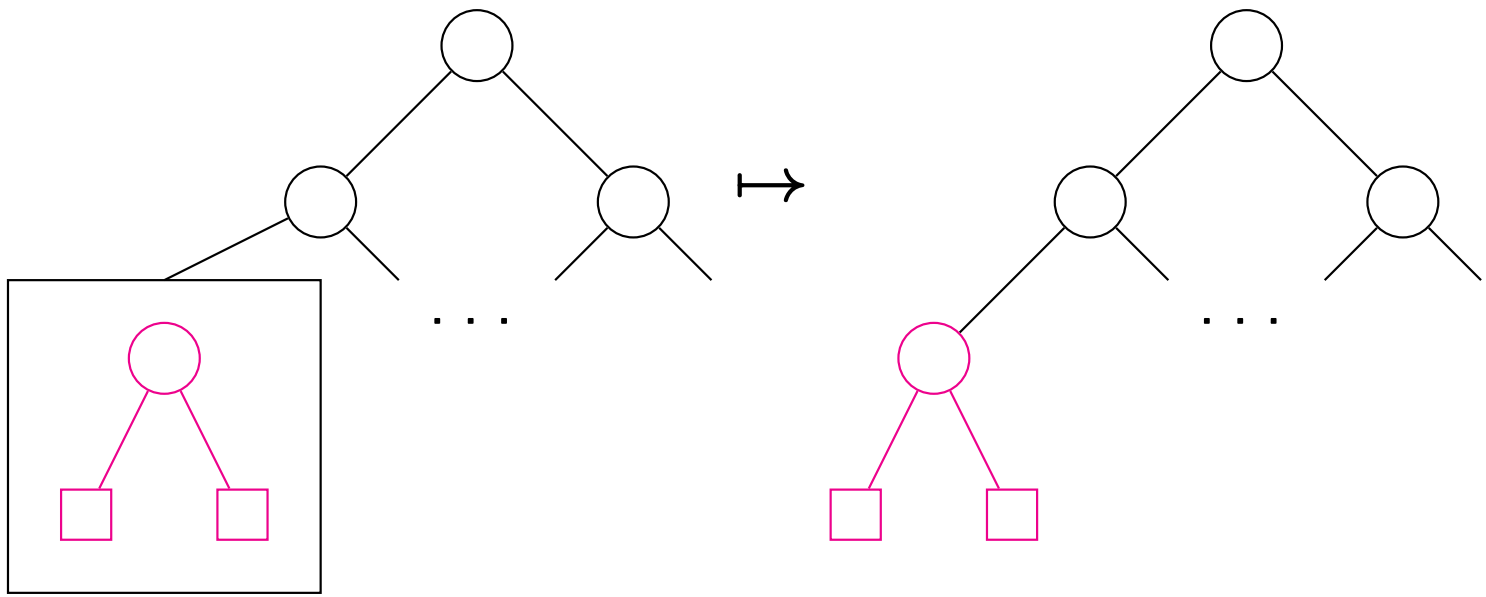
理論からできたモナド T の単位 η は，集合 X の元 x をそれのみからなる木にする．

$$
\begin{array}{ccc}
X & \xrightarrow{\eta_X} & \mathrm{Term}_{\Sigma_T}(X) \\
x & \longmapsto & \boxed{x}
\end{array}
$$

掛け算

理論からできたモナド T の掛け算 μ は，入れ子になった木を受け取り，葉の内側の木を引っこ抜いて外の木にくっつける．

$$\mathrm{Term}_{\Sigma T}(\mathrm{Term}_{\Sigma T}(X)) \xrightarrow{\mu_X} \mathrm{Term}_{\Sigma T}(X)$$



代数

モナド

プログラム

より詳細な構造へ

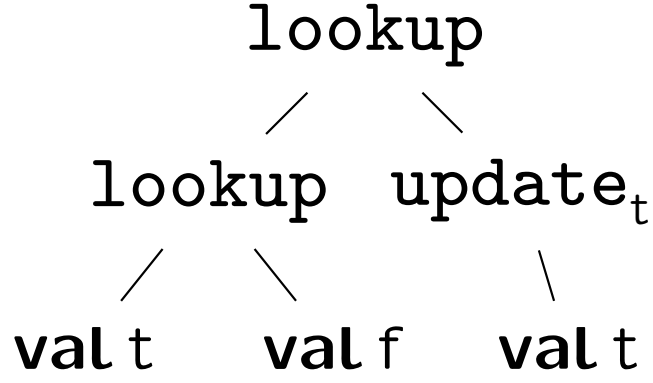
代数・モナド・プログラム

プログラムの純粋な関数としての振る舞いを壊すような挙動のことを副作用や計算効果とよぶ.

- 1990 年ごろの Moggi の研究：さまざまな計算効果がモナドによって理解できることが明らかになった.
- 2001 年の Plotkin と Power の研究：計算効果を代数の演算として理解する

計算効果の例

- 入出力
- 例外
- 状態の読み書き



プログラミング言語

文法

値：

$$V, W ::= x \mid () \mid \mathsf{t} \mid \mathsf{f} \mid \lambda x.M$$

計算：

$$\begin{aligned} M, N ::= {} & \mathbf{val}\, V \mid \sigma \mid VW \\ & \mid \mathbf{let}\, x \leftarrow M\, \mathbf{in}\, N \\ & \mid \mathbf{if}\, V\, \mathbf{then}\, M\, \mathbf{else}\, N \end{aligned}$$

ここで $\sigma \in \Sigma$**.**

$\Sigma_{\mathsf{exc}} = \{\mathtt{err}\}$ **とする.** $\mathrm{arity}(\mathtt{err}) = 0$
$\Sigma_{\mathsf{st}} = \{\mathtt{update}, \mathtt{lookup}\}$ **とする.** $\mathrm{arity}(\mathtt{update}) = 1$**,**
$\mathrm{arity}(\mathtt{lookup}) = 2$**.**

プログラムの例

例 Σ_{exc}

$$\mathbf{if}\ x\ \mathbf{then}\ \mathbf{val}\ ()\ \mathbf{else}\ \mathtt{err}$$

例 Σ_{st}

$$\begin{aligned}
&\mathbf{let}\ x \leftarrow \mathtt{lookup}\ \mathbf{in}\\
&\mathbf{if}\ x\\
&\mathbf{then}\ \mathbf{let}\ y \leftarrow \mathtt{lookup}\ \mathbf{in}\ \mathbf{val}\ y\\
&\mathbf{else}\ \mathbf{let}\ z \leftarrow \mathtt{update}_{\mathtt{t}}\ \mathbf{in}\ \mathbf{val}\ \mathtt{t}
\end{aligned}$$

型

プログラムは「情報の変換」である．どのような情報を受け取って，どのような情報を返すかを型として表現する．

型

型を次のように定義する．

$$A, B ::= \mathsf{Unit} \mid \mathsf{Bool} \mid A \to B$$

$A \to B$ は型 A のデータを型 B のデータに変換するプログラムを表す型である．

型の例

以下は型である．

$$\mathsf{Unit} \to \mathsf{Bool}, \quad (\mathsf{Bool} \to \mathsf{Unit}) \to \mathsf{Bool},$$

$$\mathsf{Unit}, \quad \mathsf{Bool} \to (\mathsf{Unit} \to \mathsf{Bool}).$$

型判断

型判断

$\Gamma = x_1 : A_1, \ldots, x_n : A_n$ とする．以下の形の記述を型判断と呼ぶ．

- 値の型判断 $\Gamma \vdash V : A$
- 計算の型判断 $\Gamma \vdash^c M : A$

- $\Gamma \vdash V : A$ は，自由変数 x_i たちがそれぞれ A_i という型を持つという状況のもとで値 V は型 A をもつという主張である．
- $\Gamma \vdash^c M : A$ は，自由変数 x_i たちがそれぞれ A_i という型を持つという状況のもとで計算 M は型 A をもつという主張である．

型判断の導出

導出規則（の一部）

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \qquad \frac{}{\Gamma \vdash () : \mathsf{Unit}} \qquad \frac{}{\Gamma \vdash \mathsf{t} : \mathsf{Bool}} \qquad \frac{}{\Gamma \vdash \mathsf{f} : \mathsf{Bool}}$$

$$\frac{\Gamma, x : A \vdash^c M : B}{\Gamma \vdash \lambda x.M : A \to B}$$

$$\frac{\Gamma \vdash V : A}{\Gamma \vdash^c \mathbf{val}\, V : A} \qquad \frac{\Gamma \vdash^c M : A \quad \Gamma, x : A \vdash^c N : B}{\Gamma \vdash^c \mathbf{let}\, x \leftarrow M\, \mathbf{in}\, N : B}$$

$$\frac{\sigma \in \Sigma \quad \mathrm{arity}(\sigma) = 2}{\Gamma \vdash^c \sigma : \mathsf{Bool}} \qquad \frac{\sigma \in \Sigma \quad \mathrm{arity}(\sigma) = 1}{\Gamma \vdash^c \sigma : \mathsf{Unit}}$$

$$\frac{\Gamma \vdash V : A \to B \quad \Gamma \vdash W : A}{\Gamma \vdash^c VW : B}$$

プログラムの意味

プログラムは構文的な対象である．数学の構造を使ってプログラムに意味を与えることで，プログラムの性質を数学的に議論できるようにしたい．

文字列としてのプログラム	$\to$	数学的な構造
$\Gamma \vdash V : A$	$\mapsto$	$[\![\Gamma]\!] \xrightarrow{[\![V]\!]} [\![A]\!]$
$\Gamma \vdash^c M : A$	$\mapsto$	$[\![\Gamma]\!] \xrightarrow{[\![M]\!]} T[\![A]\!]$

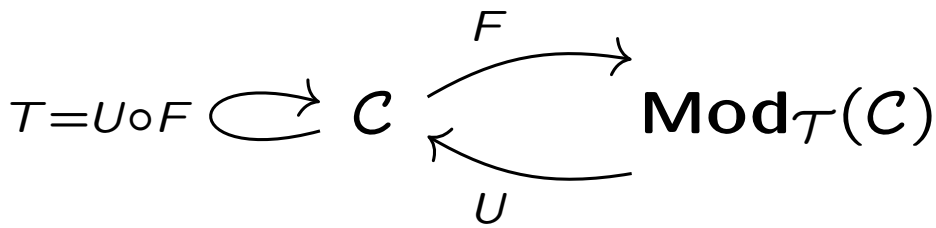
$\mathcal{C}$ を有限積とべきを持つ圏とする．

$$\mathcal{C}(A \times B, C) \cong \mathcal{C}(A, C^B)$$

特に Set は有限積とべきを持つ．

解釈のための構造

以下の随伴 $F \dashv U$ があるとする．



イメージとしては，値は圏 $\mathcal{C}$ で解釈され，計算は圏 $\mathbf{Mod}_T(\mathcal{C})$ で解釈される．簡単のため以降では $\mathcal{C} = \mathbf{Set}$ とする．

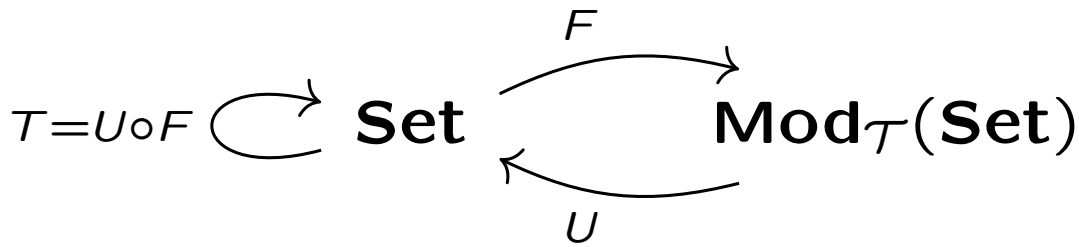
型の解釈

型の解釈

型 A の解釈 $[\![A]\!]$ を Set の対象として定める．

- $[\![\mathsf{Unit}]\!] = 1 \in \mathbf{Set}$
- $[\![\mathsf{Bool}]\!] = 2 \in \mathbf{Set}$
- $[\![A \to B]\!] = [\![B]\!]^{[\![A]\!]} \in \mathbf{Set}$

プログラムの解釈



$\Gamma = x_1 : A_1, \ldots, x_n : A_n$ **のとき，**
$[\![\Gamma]\!] = [\![A_1]\!] \times \cdots \times [\![A_n]\!]$ **とする．導出可能な型判断に対して**
その解釈を次のように $\mathcal{C} = \mathbf{Set}$ **の射として定めたい．**

- $[\![\Gamma \vdash V : A]\!] : [\![\Gamma]\!] \to [\![A]\!]$
- $[\![\Gamma \vdash^c M : A]\!] : [\![\Gamma]\!] \to T[\![A]\!]$

値の解釈

値の解釈

$$[\![\Gamma \vdash x_i : A_i]\!] = \left([\![\Gamma]\!] \xrightarrow{\pi_i} [\![A_i]\!] \right)$$

$$[\![\Gamma \vdash () : \mathsf{Unit}]\!] = \left([\![\Gamma]\!] \xrightarrow{!_\Gamma} 1 \right)$$

$$[\![\Gamma \vdash \mathsf{t} : \mathsf{Bool}]\!] = \left([\![\Gamma]\!] \xrightarrow{!_\Gamma} 1 \xrightarrow{[\![\mathsf{t}]\!]} [\![\mathsf{Bool}]\!] \right)$$

$$[\![\Gamma \vdash \mathsf{f} : \mathsf{Bool}]\!] = \left([\![\Gamma]\!] \xrightarrow{!_\Gamma} 1 \xrightarrow{[\![\mathsf{f}]\!]} [\![\mathsf{Bool}]\!] \right)$$

$$[\![\Gamma \vdash \lambda x.M : A \to B]\!] = \left([\![\Gamma]\!] \xrightarrow{\Lambda[\![M]\!]} (T[\![B]\!])^{[\![A]\!]} \right)$$

計算の解釈

計算の解釈

$$[\![\Gamma \vdash^c \mathbf{val}\, V : A]\!] = \left([\![\Gamma]\!] \xrightarrow{[\![V]\!]} [\![A]\!] \xrightarrow{\eta_A} T[\![A]\!] \right)$$

$$[\![\Gamma \vdash^c \sigma : \mathsf{Bool}]\!]$$
$$= \left([\![\Gamma]\!] \xrightarrow{!} 1 \xrightarrow{\sigma(\Lambda[\![x{:}\mathsf{Bool} \vdash^c \mathbf{val}\, x{:}\mathsf{Bool}]\!])} T[\![\mathsf{Bool}]\!] \right)$$

$$[\![\Gamma \vdash^c \mathbf{let}\, x \leftarrow M\, \mathbf{in}\, N : B]\!]$$
$$= \left(\begin{array}{l} [\![\Gamma]\!] \xrightarrow{\langle [\![\Gamma]\!], [\![M]\!] \rangle} [\![\Gamma]\!] \times T[\![A]\!] \xrightarrow{\mathrm{st}} T([\![\Gamma]\!] \times [\![A]\!]) \\ \xrightarrow{T[\![N]\!]} TT[\![B]\!] \xrightarrow{\mu_{[\![B]\!]}} T[\![B]\!] \end{array} \right)$$

単位と掛け算

val V **はモナドの単位に対応：**

$$\frac{\vdash V : A}{\vdash^c \mathbf{val}\, V : A} \qquad \frac{1 \xrightarrow{[\![V]\!]} [\![A]\!]}{1 \xrightarrow{[\![V]\!]} [\![A]\!] \xrightarrow{\eta_{[\![A]\!]}} T[\![A]\!]}$$

let $x \leftarrow M$ **in** N **はモナドの掛け算に対応：**

$$\frac{\vdash^c M : A \quad x : A \vdash^c N : B}{\vdash^c \mathbf{let}\, x \leftarrow M\, \mathbf{in}\, N : B}$$

$$\frac{1 \xrightarrow{[\![M]\!]} T[\![A]\!] \quad [\![A]\!] \xrightarrow{[\![N]\!]} T[\![B]\!]}{1 \xrightarrow{[\![M]\!]} T[\![B]\!] \xrightarrow{T[\![N]\!]} TT[\![B]\!] \xrightarrow{\mu_{[\![B]\!]}} T[\![B]\!]}$$

プログラムの解釈の例

$$
\left[\!\left[
\begin{array}{l}
\mathbf{let}\, x \leftarrow \mathtt{lookup}\, \mathbf{in} \\
\mathbf{if}\, x \\
\mathbf{then}\, \mathbf{let}\, y \leftarrow \mathtt{lookup}\, \mathbf{in}\, \mathbf{val}\, y \\
\mathbf{else}\, \mathbf{let}\, z \leftarrow \mathtt{update}_{\mathtt{t}}\, \mathbf{in}\, \mathbf{val}\, \mathtt{t}
\end{array}
\right]\!\right]
= \left(
\begin{array}{ccc}
 & \mathtt{lookup} & \\
 \swarrow & & \searrow \\
\mathtt{lookup} & & \mathtt{update}_{\mathtt{t}} \\
\swarrow \quad \searrow & & \quad \searrow \\
\mathbf{val}\, \mathtt{t} & \mathbf{val}\, \mathtt{f} & \mathbf{val}\, \mathtt{t}
\end{array}
\right) : 1 \to T[\![\mathtt{Bool}]\!]
$$

ハンドラ

まずは最もよく知られた例である例外ハンドラを見てみよう.

handle
  エラーが起きそうな処理 M
with{
  val $x \mapsto A$ でエラーが起きなかった場合の処理 D
  $\mathtt{err} \mapsto A$ でエラー $\mathtt{err}$ が起きた場合の処理 E
}

よくあるプログラミング言語では

$$\mathbf{try}\,\{M\}\,\mathbf{catch}(\mathtt{err})\,\{E\}$$

のような構文であることが多い.

ハンドラの観察

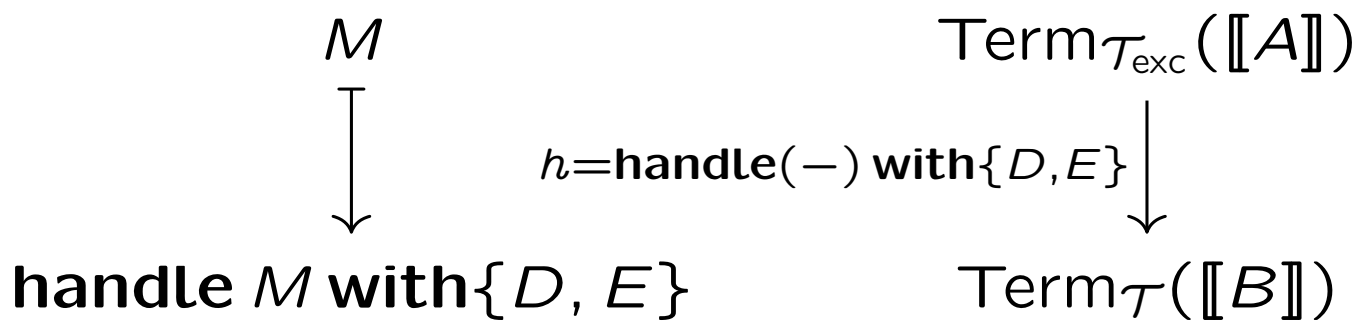
- M **は例外の理論** $\mathcal{T}_{\mathrm{exc}} = (\{\mathtt{err}\}, \varnothing)$ **の自由代数** $\mathrm{Term}_{\mathcal{T}_{\mathrm{exc}}}(\llbracket A \rrbracket)$ **で解釈される．**
- D **と** E **は例外が起き得ないとする．つまり** $\mathtt{err}$ **をシグネチャに含まない理論** $\mathcal{T}$ **の自由代数** $\mathrm{Term}_{\mathcal{T}}(\llbracket B \rrbracket)$ **で解釈される．**

handle エラーが起きそうな処理 $M : A$
with{
  val $x \mapsto A$ **でエラーが起きなかった場合の処理** $D : B$
  $\mathtt{err} \mapsto A$ **でエラー** $\mathtt{err}$ **が起きた場合の処理** $E : B$ }

上のハンドラは自由 $\mathcal{T}_{\mathrm{exc}}$ **代数を** $\mathcal{T}$ **代数に変換している．**

自由代数の普遍性とハンドラ

ハンドラは以下のような「射」になってほしい．



Q. **これを何らかの意味で正当化できるか？**

A. **ハンドラ** h **は自由代数** $\mathsf{Term}_{\mathcal{T}_{\mathrm{exc}}}([\![A]\!])$ **からの普遍性による** $\mathcal{T}_{\mathrm{exc}}$ **代数の射とみなせる．**

そのためには $[\![B]\!]$ **上自由生成された** $\mathcal{T}$ **代数** $\mathsf{Term}_{\mathcal{T}}([\![B]\!])$ **を** $[\![A]\!]$ **上の** $\mathcal{T}_{\mathrm{exc}}$ **代数とみなせればよい．**

ハンドラのレシピ

$\mathsf{Term}_{\mathcal{T}}(\llbracket B \rrbracket)$ **を** $\llbracket A \rrbracket$ **上の** $\mathcal{T}_{\mathrm{exc}}$ **代数とみなして，**
$\mathsf{Term}\,\mathcal{T}_{\mathrm{exc}}(\llbracket A \rrbracket)$ **からの普遍性による射を作りたい．そのためには以下のものがあればよい．**

1. **演算** $\mathtt{err}$ **の** $\mathsf{Term}_{\mathcal{T}}(\llbracket B \rrbracket)$ **における解釈**
   $|\mathtt{err}| : 1 \to \mathsf{Term}_{\mathcal{T}}(\llbracket B \rrbracket)$
2. **射** $\llbracket A \rrbracket \to U\,\mathsf{Term}_{\mathcal{T}}(\llbracket B \rrbracket)$

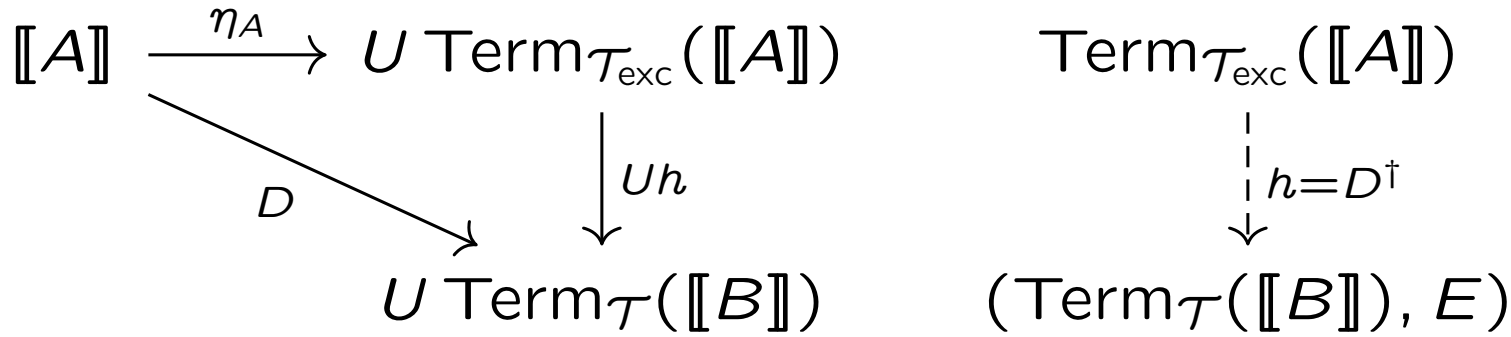
そしてこれらはすでにある．

handle$(-)$ **with**$\{$
  val $x \mapsto D : B$
  $\mathtt{err} \mapsto E : B$
$\}$

1. E **は** $\mathtt{err}$ **の解釈であり，これにより** $(\mathsf{Term}_{\mathcal{T}}(\llbracket B \rrbracket), E)$ **は** $\mathcal{T}_{\mathrm{exc}}$ **代数とみなせる．**
2. D **は射** $\llbracket A \rrbracket \to U\,\mathsf{Term}_{\mathcal{T}}(\llbracket B \rrbracket)$ **である．**

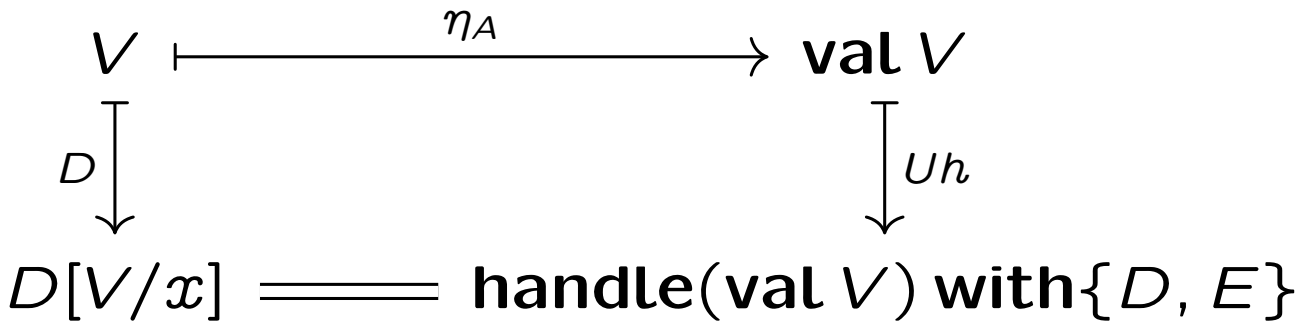
例外ハンドラは自由 $\mathcal{T}_{\mathrm{exc}}$ 代数からの普遍性による射

図式で表すとこうなる．



上の三角形の可換図式は以下の等式を主張している．

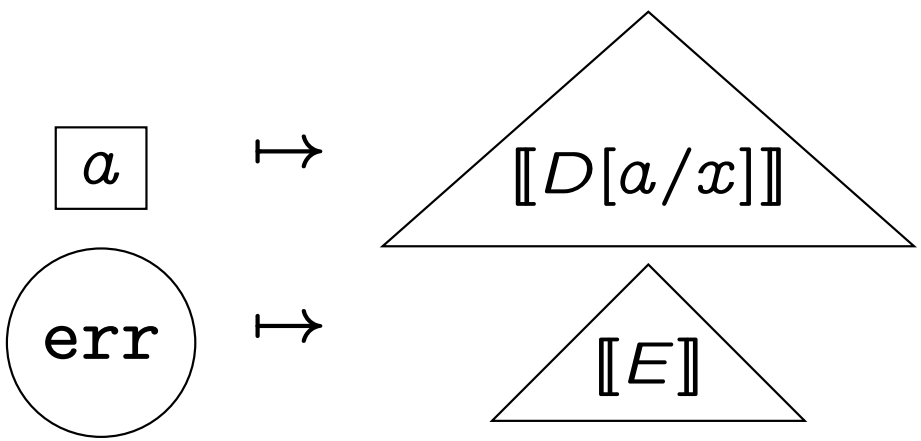
$$D[V/x] = \mathbf{handle}(\mathbf{val}\, V)\, \mathbf{with}\{\mathbf{val}\, x \mapsto D, \mathtt{err} \mapsto E\}$$



木の変換としてのハンドラ

$\mathsf{Term}_{\mathcal{T}}(X)$ **の元は木とみなせるのであった．そこでハンドラを木の変換として観察してみよう．**

$$\mathsf{Term}_{\mathcal{T}_{\mathsf{exc}}}(A) \xrightarrow{h} \mathsf{Term}_{\mathcal{T}}(B)$$



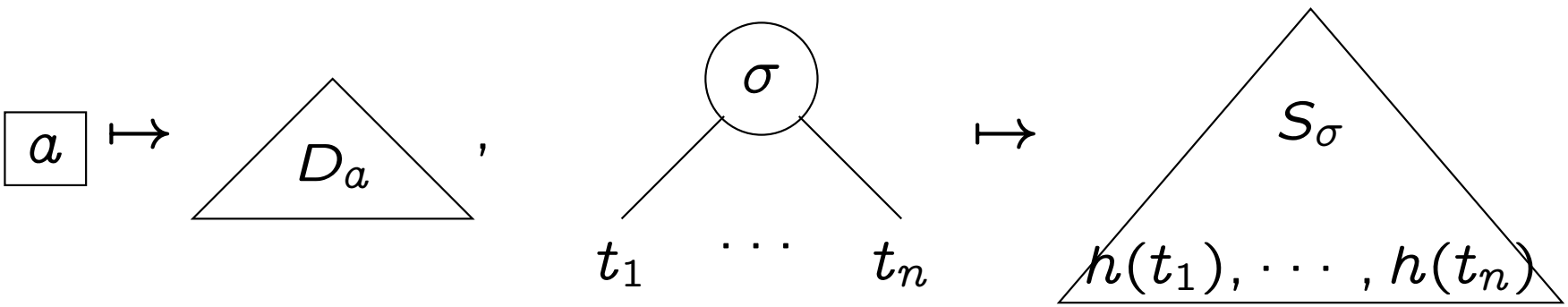
一般のエフェクトハンドラ

自由 $\mathcal{T}$ 代数からの普遍性による射として一般のエフェクトハンドラを構成できる．今，理論 $\mathcal{T}$，$\mathcal{T}'$ があるとする．木の変換 h

$$\mathrm{Term}_{\mathcal{T}}(A) \xrightarrow{h} \mathrm{Term}_{\mathcal{T}'}(B)$$

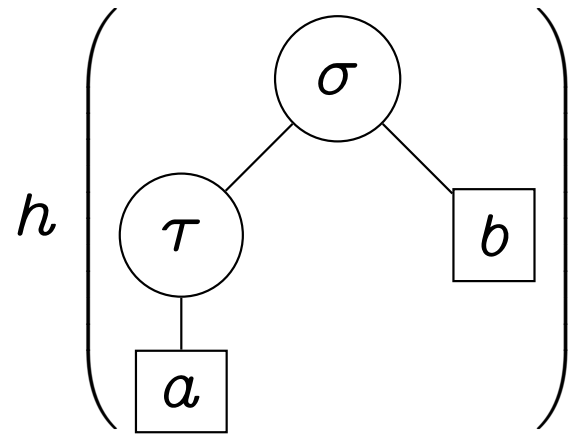
を作るには以下の情報があればよい．

1. **各 $a \in A$ に対して，木 $D_a \in \mathrm{Term}_{\mathcal{T}'}(B)$**
2. **各演算 $\sigma \in \Sigma_{\mathcal{T}}$（引数 $n = \mathrm{arity}(\sigma)$）に対して，$n$ 個の $\mathrm{Term}_{\mathcal{T}'}(B)$ の木が与えられたときに $\mathrm{Term}_{\mathcal{T}'}(B)$ の木を与える変換 $S_\sigma(t'_1, \ldots, t'_n)$**



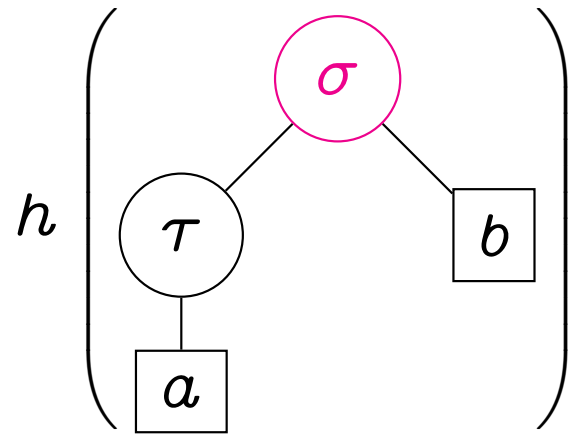
木の変換の例

$$\mathrm{Term}_{\mathcal{T}}(A) \xrightarrow{h} \mathrm{Term}_{\mathcal{T}'}(B)$$



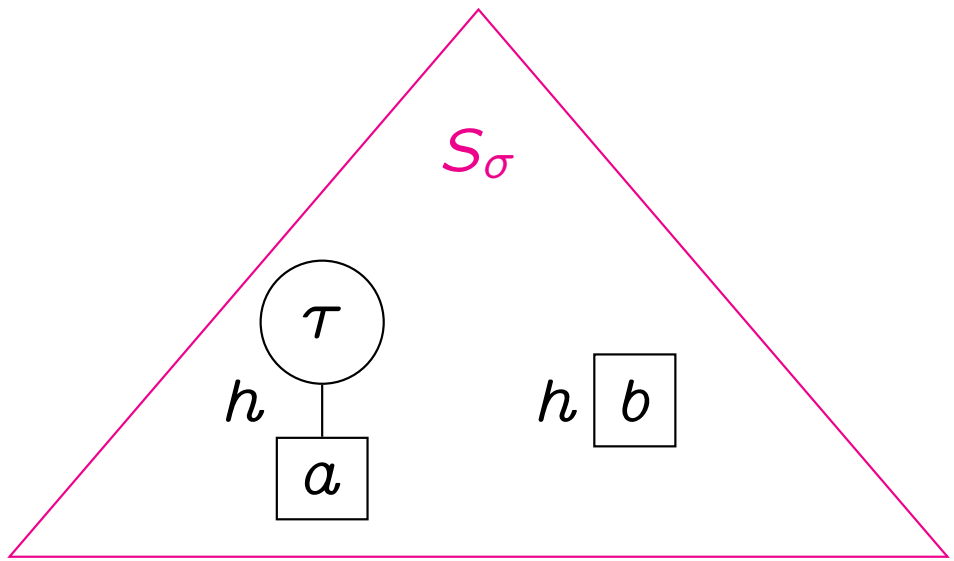
木の変換の例

$$\mathrm{Term}_{\mathcal{T}}(A) \xrightarrow{h} \mathrm{Term}_{\mathcal{T}'}(B)$$



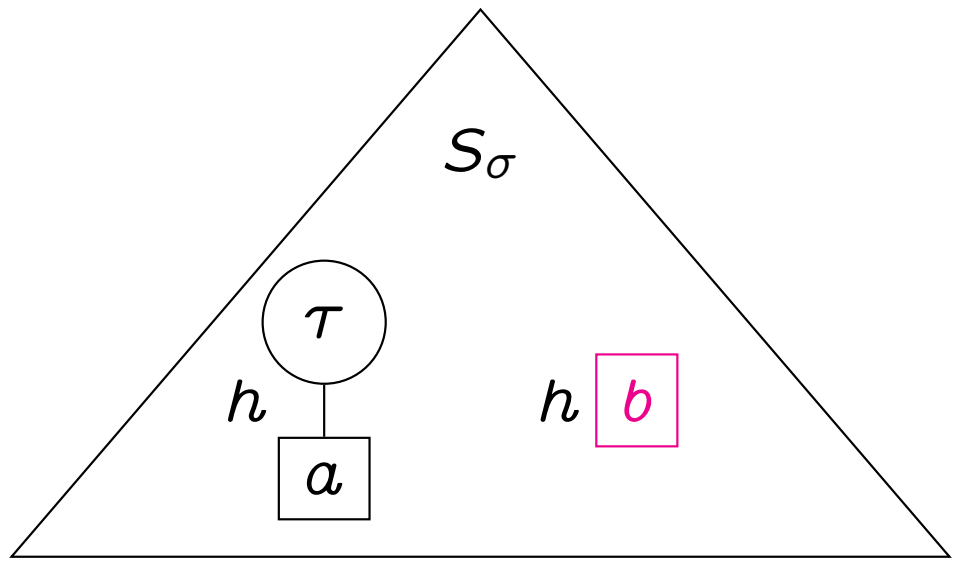
木の変換の例

$$\mathrm{Term}_{\mathcal{T}}(A) \xrightarrow{h} \mathrm{Term}_{\mathcal{T}'}(B)$$



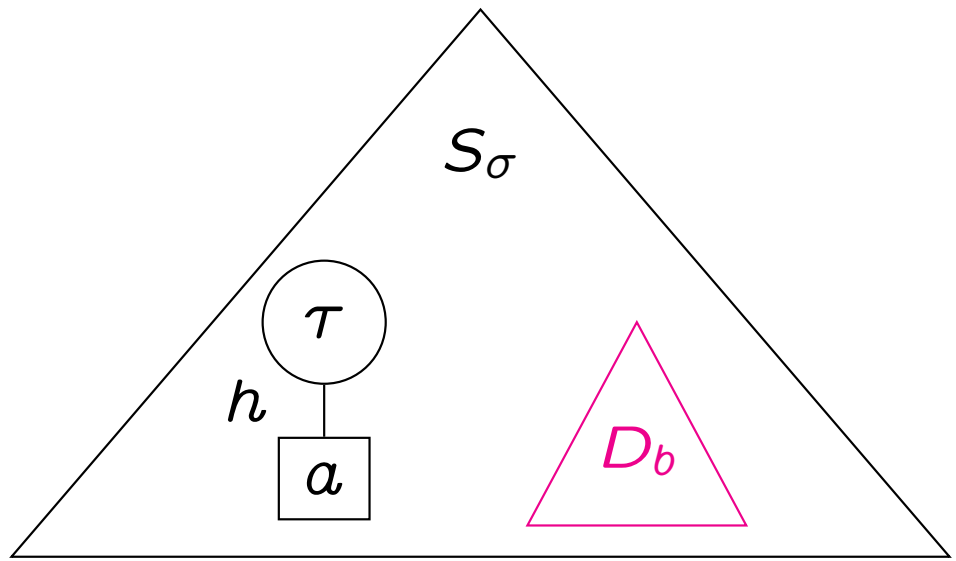
木の変換の例

$$\mathrm{Term}_{\mathcal{T}}(A) \xrightarrow{h} \mathrm{Term}_{\mathcal{T}'}(B)$$



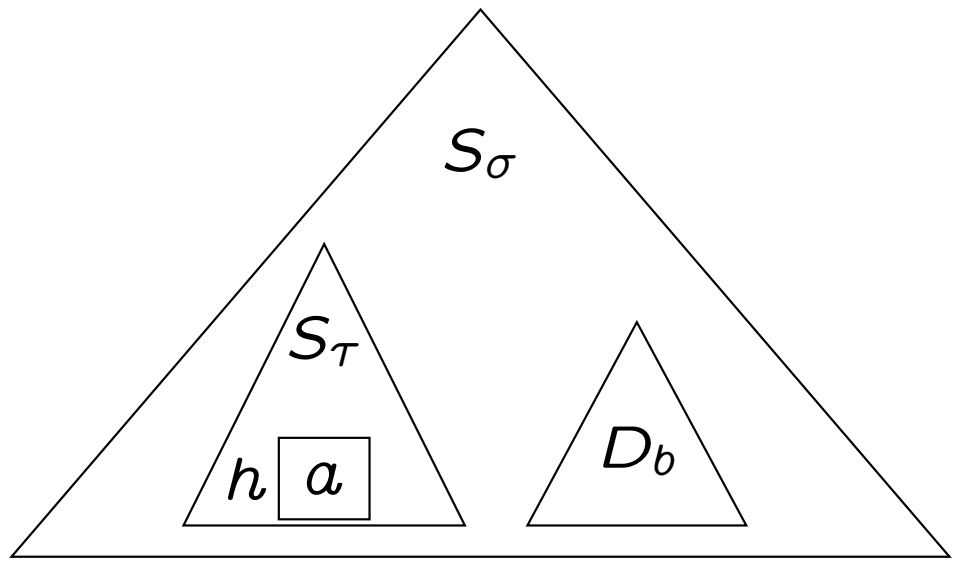
木の変換の例

$$\mathrm{Term}_{\mathcal{T}}(A) \xrightarrow{h} \mathrm{Term}_{\mathcal{T}'}(B)$$



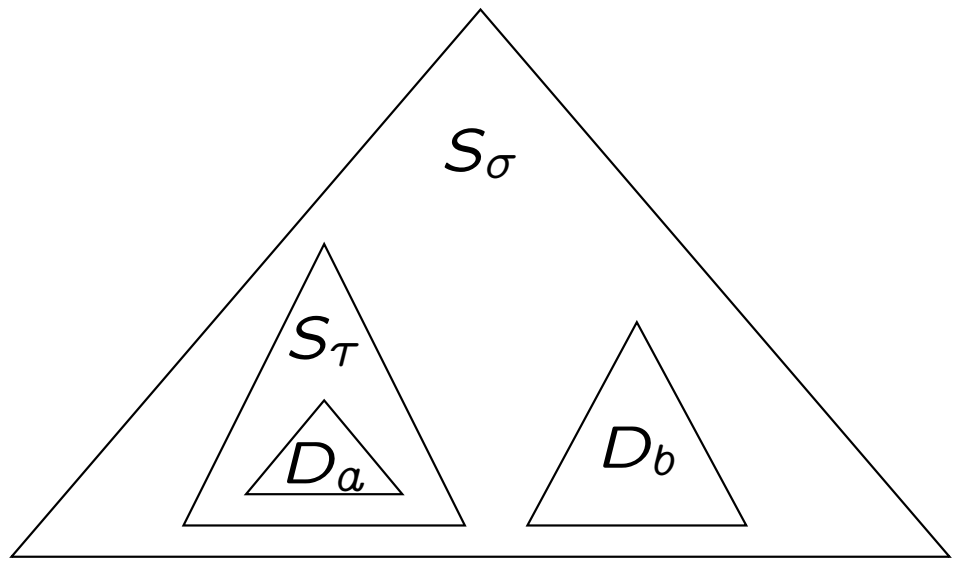
木の変換の例

$$\mathrm{Term}_{\mathcal{T}}(A) \xrightarrow{h} \mathrm{Term}_{\mathcal{T}'}(B)$$



木の変換の例

$$\mathrm{Term}_{\mathcal{T}}(A) \xrightarrow{h} \mathrm{Term}_{\mathcal{T}'}(B)$$



一般のエフェクトハンドラの構文

$$
\begin{array}{l}
\textbf{handle}\ M\ \textbf{with}\{ \\
\quad \textbf{val}\ x \mapsto D; \\
\quad \sigma, k \mapsto S_\sigma \\
\}
\end{array}
$$

k は M で σ が生じたときの継続（残りの計算）である．これは S_σ に与えられる $\mathrm{arity}(\sigma)$ 個の木たち $(h(t_1), \ldots, h(t_{\mathrm{arity}(n)}))$ を表す．

代数

モナド

プログラム

より詳細な構造へ

モナドの拡張

モナドは

- $\mathcal{C}$ **上の自己関手の圏** $[\mathcal{C}, \mathcal{C}]$ **のモノイド対象である**
- **対象がただひとつの** $[\mathcal{C}, \mathcal{C}]$**-圏である**
- **自明な圏** $\mathbf{1}$ **から** $\mathrm{Endo}(\mathcal{C})$ **への緩関手** $\mathbf{1} \xrightarrow{\mathrm{lax}} \mathrm{Endo}(\mathcal{C})$ **である**

一番最後の見方を採用しよう．自明な圏 $\mathbf{1}$ **を一般の圏** $\mathcal{S}^{\mathrm{op}}$ **で置き換えることでモナドを拡張した概念が得られる．**

次数圏付きモナド [Orchard, Wadler, Eades III. 2020]

$\mathcal{S}$**-次数圏付きモナドとは** $\mathcal{S}^{\mathrm{op}}$ **から** $\mathrm{Endo}(\mathcal{C})$ **への緩関手**

$\mathcal{S}^{\mathrm{op}} \xrightarrow{\mathrm{lax}} \mathrm{Endo}(\mathcal{C})$ **である**

次数圏付き代数理論

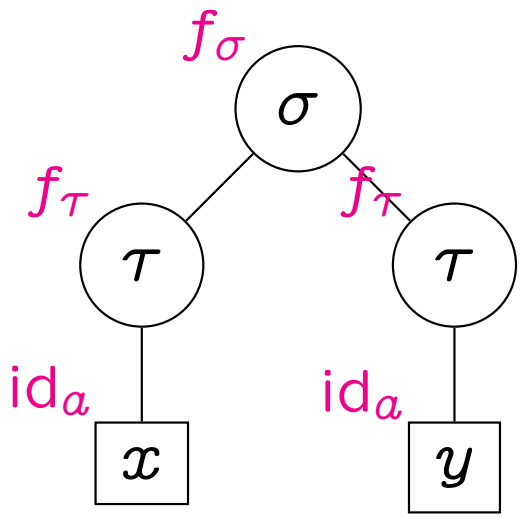
モナドと代数は対応しているのであった．

Q. **では $\mathcal{S}$-次数圏付きモナド $\mathcal{S}^{\mathrm{op}} \xrightarrow{\mathrm{lax}} \mathrm{Endo}(\mathcal{C})$ に対応する代数の概念は作れるだろうか？**

A. **作れる** [S. 2021]**．さらに対応するプログラミング言語も作れる．これは通信のプロトコルに従っていることを静的に検査するのに便利．また次数圏付きエフェクトのエフェクトハンドラも作れる．**

次数圏付き項のイメージ

$\Sigma = \{\sigma, \tau, \dots\}$, $X = \{x, y, z, \dots\}$ **とする．各演算には** S **の射が割り当てられている．**



$\in \mathrm{Term}_\Sigma(f_\tau \circ f_\sigma, X)$

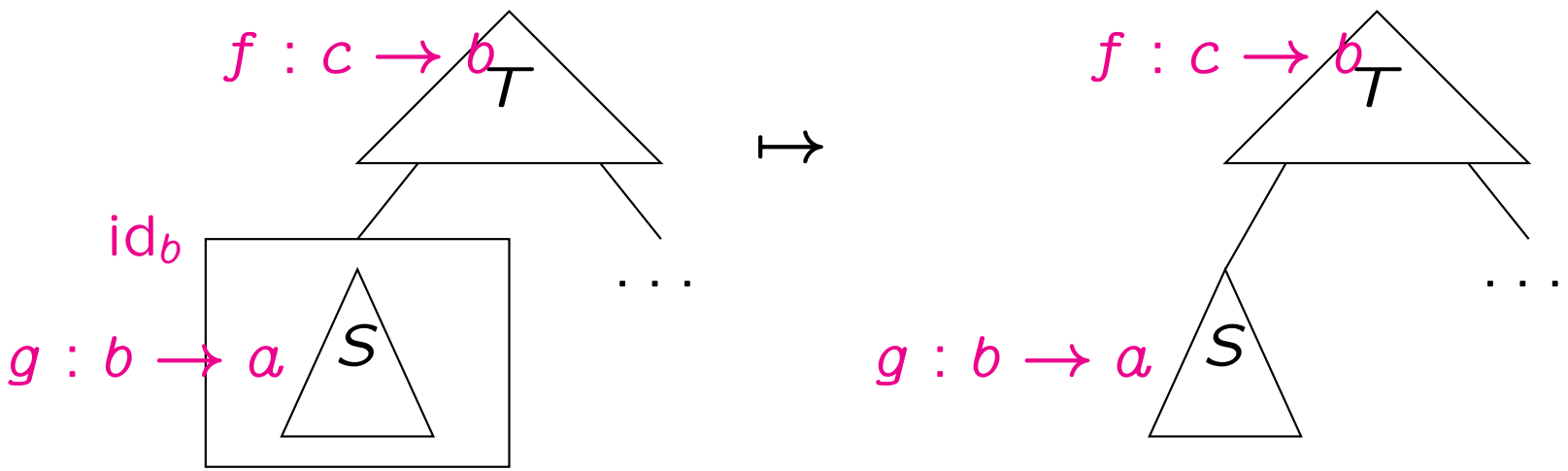
S **の射** f **で次数付けられた項は，葉が** X**，中間ノードが** Σ **でラベル付けられた木で，かつ根から葉まで演算に割り当てられた射を合成すると** f **になるようなもの．**

次数圏付き代数の単位と掛け算

$$X \xrightarrow{\eta^a_X} \mathrm{Term}_\Sigma(\mathrm{id}_a, X)$$

$$x \longmapsto \boxed{x}^{\mathrm{id}_a}$$

$$\mathrm{Term}_\Sigma(f, \mathrm{Term}_\Sigma(g, X)) \xrightarrow{\mu^{f,g}_X} \mathrm{Term}_\sigma(g \circ f, X)$$



まとめ

- 代数からモナドが作れる
- 代数はプログラムの解釈に使える
- ハンドラは自由代数からの普遍性による射である

これを次数圏付き版に拡張して，

- 次数圏付き代数・次数圏付きモナド・次数圏付きプログラム

の関係が得られる．