# プログラム意味論とファイブレーション(仮)

真田 嵩大 数理解析研究所 D1

Yota25,2022年3月26日

#### Contents

ファイブレーション

プログラム意味論

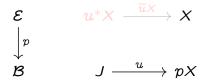
プログラム意味論とファイブレーション

プログラム意味論

プログラム意味論とファイブレーション

**▶** E, B: 圏

p: E → B: 関手

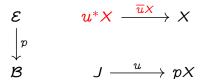


任意の対象  $J \in \mathcal{B}$  ,  $X \in \mathcal{E}$  と射  $u: J \to pX$  に対して , ある対象  $u^*X \in \mathcal{E}$  と射  $\overline{u}X: u^*X \to X$  が存在して  $p(\overline{u}X) = u$  である . さらに  $\overline{u}X$  は次の普遍性を持つ .

E,B: 圏

p: E → B: 関手

 $p: \mathcal{E} \to \mathcal{B}$  がファイブレーションであるとは,

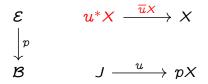


任意の対象  $J \in \mathcal{B}$  ,  $X \in \mathcal{E}$  と射  $u: J \to pX$  に対して,ある対象  $u^*X \in \mathcal{E}$  と射  $\overline{u}X: u^*X \to X$  が存在して  $p(\overline{u}X) = u$  である.さらに  $\overline{u}X$  は次の普遍性を持つ.

**▶** E , B: 圏

p: E → B: 関手

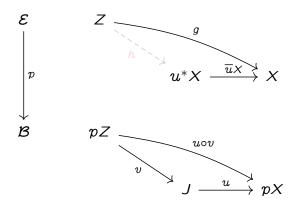
 $p: \mathcal{E} \to \mathcal{B}$  がファイブレーションであるとは,



任意の対象  $J \in \mathcal{B}$  ,  $X \in \mathcal{E}$  と射  $u: J \to pX$  に対して , ある対象  $u^*X \in \mathcal{E}$  と射  $\overline{u}X: u^*X \to X$  が存在して  $p(\overline{u}X) = u$  である . さらに  $\overline{u}X$  は次の普遍性を持つ .

### カルテシアン射の普遍性

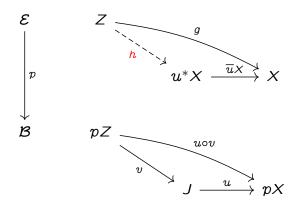
 $\overline{u}X$  は次の普遍性を持つ.



任意の  $Z \in \mathcal{E} \succeq g: Z \to X$ ,  $v: pZ \to J$  に対して  $p(g) = u \circ v$  ならば, p(h) = v を満たす  $h: Z \to u^*X$  が 一意的に存在する.このような普遍性を持つ射をカルテシアン 射と呼ぶ.

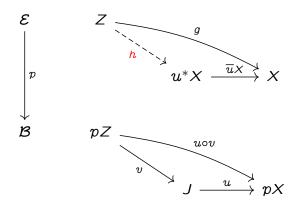
### カルテシアン射の普遍性

 $\overline{u}X$  は次の普遍性を持つ.



#### カルテシアン射の普遍性

 $\overline{u}X$  は次の普遍性を持つ.



### ファイブレーションの気持ち

ファイブレーション  $p: \mathcal{E} \to \mathcal{B}$  を以下のような気持ちで眺める.

- ▶ Bは基本的な構造の圏
- ε は β により細かな構造が付加された圏
- ▶  $p \bowtie X \in \mathcal{E}$  に対してそのベースの構造  $pX \in \mathcal{B}$  を抽出する
- ▶  $u:J\to pX$  に対して  $u^*X\in\mathcal{E}$  は , u を  $\mathcal{E}$  に持ち上げられるような最も弱い構造

### ファイブレーションの具体例: Pred $\rightarrow$ Set

Set を集合と写像の圏とする、圏 Pred を次で定める、

- ▶ 対象は組 (I, X) たち.ここで  $I \in \mathbf{Set}$  かつ  $X \subset I$  .
- ▶ 射  $f:(J,Y) \to (I,X)$  は写像  $f:J \to I$  であって Y の元 を X にうつすもの .

関手p: Pred  $\rightarrow$  Set を

$$(I,X) \mapsto I$$
$$f \mapsto f$$

と定める、このとき p はファイブレーションとなる、 $u: J \to p(I, X)$  に対して

$$u^*(I, X) = (J, u^{-1}(X)) = (J, \{j \in J \mid uj \in X\})$$

である、このことから  $u^*$  のことを逆像関手と呼ぶ、

### ファイブレーションの具体例: Pred $\rightarrow$ Set

Set を集合と写像の圏とする、圏 Pred を次で定める、

- ▶ 対象は組 (I, X) たち.ここで  $I \in \mathbf{Set}$  かつ  $X \subseteq I$  .
- ▶ 射  $f:(J,Y) \to (I,X)$  は写像  $f:J \to I$  であって Y の元を X にうつすもの .

関手p: Pred  $\rightarrow$  Set を

$$(I,X) \mapsto I$$
$$f \mapsto f$$

と定める、このときpはファイブレーションとなる。

$$u: J \to p(I,X)$$
 に対して

$$u^*(I, X) = (J, u^{-1}(X)) = (J, \{j \in J \mid uj \in X\})$$

である、このことから  $u^*$  のことを逆像関手と呼ぶ

### ファイブレーションの具体例: Pred $\rightarrow$ Set

Set を集合と写像の圏とする、圏 Pred を次で定める、

- ▶ 対象は組 (I, X) たち.ここで  $I \in \mathbf{Set}$  かつ  $X \subseteq I$  .
- ▶ 射  $f:(J,Y) \to (I,X)$  は写像  $f:J \to I$  であって Y の元を X にうつすもの .

関手p: Pred  $\rightarrow$  Set を

$$(I,X) \mapsto I$$
$$f \mapsto f$$

と定める.このとき p はファイブレーションとなる.  $u: J \to p(I, X)$  に対して

$$u^*(I, X) = (J, u^{-1}(X)) = (J, \{j \in J \mid uj \in X\})$$

である、このことから  $u^*$  のことを逆像関手と呼ぶ、

#### プログラム意味論

プログラム意味論とファイブレーション

## プログラム意味論

単なる文字列としてのプログラムに何らか意味を与えることで計 算の性質を研究する.

- ▶ 操作的意味論:プログラムをどう実行するかを指定する(= 簡約規則を与える)ことで意味を定める
- ▶ 表示的意味論:プログラムが何を表すかを指定する(=プログラムの集合から数学的構造への割り当てを定める)ことで意味を定める.

今回は表示的意味論,特にプログラムの意味を圏によって解釈する<mark>圏論的意味論</mark>について考える.

### 型とプログラム

データの種類のことを型と呼ぶ.プログラムはデータを加工する レシピである.一般に型がついたプログラムを以下のように書く.

 $x_1:\tau_1,\ldots,x_n:\tau_n\vdash M:\tau$ 

#### プログラムの例

Intを整数を表す型とする.

- $x: Int, y: Int \vdash add(x, y): Int は, Int 型の変数 <math>x, y$  が与えられている状況で add(x, y) というプログラムは Int 型であることを主張している.
- $ightharpoonup x: Int \vdash \lambda y^{Int}.add(x,y): Int \rightarrow Int$

### プログラムの解釈

プログラムに圏 Set における解釈を与えたい.

ightharpoonup 型 au に  $\mathbf{Set}$  の対象 , つまり集合  $[\![ au]\!]$  を割り当てる .

$$\begin{aligned}
&\llbracket \mathsf{Unit} \rrbracket = \{*\} \\
&\llbracket \mathsf{Int} \rrbracket = \mathbb{N} \\
&\llbracket \tau_1 \to \tau_2 \rrbracket = \llbracket \tau_2 \rrbracket^{\llbracket \tau_1 \rrbracket}
\end{aligned}$$

ightharpoonup プログラム  $x_1: au_1,\ldots,x_n: au_ndash M: au$  は f Set における射

$$\llbracket M \rrbracket : \llbracket \tau_1 \rrbracket \times \cdots \times \llbracket \tau_n \rrbracket \rightarrow \llbracket \tau \rrbracket$$

として解釈する.

ここでは **Set** における解釈しか考えないが,一般には適当な構造(カルテシアン閉,モノイダル閉など)を持つ圏でプログラムを解釈できる.

### プログラムの解釈の例

 $ightharpoonup x: ext{Int, } y: ext{Int } 
ightharpoonup ext{add}(x,y): ext{Int } の解釈は$ 

$$\llbracket \operatorname{add}(x,y) \rrbracket : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$
 
$$(m,n) \mapsto m+n$$

▶  $x : Int \vdash \lambda y^{Int}.add(x, y) : Int \rightarrow Int の解釈は$ 

$$\llbracket \lambda y^{ ext{Int}}.\mathsf{add}(x,y)
rbracket : \mathbb{N} o \mathbb{N}^{\mathbb{N}} \ m \mapsto (n \mapsto m+n)$$

プログラム意味論

プログラム意味論とファイブレーション

## ファイブレーションの気持ち(改)

ファイブレーション  $p: \mathcal{E} \to \mathcal{B}$  を以下のような気持ちで眺める.

- ▶ Bは基本的な構造の圏
  - ▶ 型とプログラムが解釈される圏
- ε は β により細かな構造が付加された圏
  - ▶ 型とプログラムの性質の圏
- ▶ pは  $X \in \mathcal{E}$  に対してそのベースの構造  $pX \in \mathcal{B}$  を抽出する
  - ▶ 性質を記述する領域の抽出
- ▶  $u:J\to pX$  に対して  $u^*X\in\mathcal{E}$  は , u を  $\mathcal{E}$  に持ち上げられるような最も弱い構造
  - ▶ プログラムの最弱事前条件

# プログラムのファイブレーション意味論

プログラム  $x_1:\tau_1,\ldots,x_n:\tau_n\vdash M:\tau$  に対して

 $x_1 : \tau_1, \ldots, x_n : \tau_n \{P\} \vdash M : \tau \{v.Q\}$ 

で「プログラム実行前に条件  $\frac{P}{P}$  が満たされているとき,M を実行した結果を  $\frac{v}{P}$  とすると,実行後に条件  $\frac{Q}{Q}$  が成立する」ことを表す.

これをファイブレーションを用いて以下のように表す!



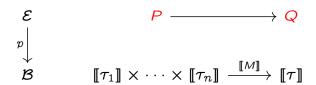
# プログラムのファイブレーション意味論

プログラム  $x_1:\tau_1,\ldots,x_n:\tau_n\vdash M:\tau$  に対して

$$x_1 : \tau_1, \ldots, x_n : \tau_n \{P\} \vdash M : \tau \{v.Q\}$$

で「プログラム実行前に条件  $\stackrel{\textbf{P}}{P}$  が満たされているとき , M を実行した結果を  $\stackrel{\textbf{v}}{V}$  とすると , 実行後に条件  $\stackrel{\textbf{Q}}{Q}$  が成立する 」ことを表す .

これをファイブレーションを用いて以下のように表す.



# ファイブレーションを使ったプログラムの解釈の具体例

変数  $x \ge y$  が x = 0 かつ y = 2 という条件を満たしているならば,プログラム add(x, y) を実行した結果の値は偶数である.

$$x: \text{Int}, y: \text{Int } \{x = 0 \land y = 2\}$$
  
  $\vdash \text{add}(x, y): \text{Int } \{v.v \text{ is even}\}$ 

この主張が正しいことは,プログラム  $\operatorname{add}(x,y)$  の  $\operatorname{Set}$  における解釈がファイブレーション  $\operatorname{p}$  に沿って  $\operatorname{Pred}$  に持ち上がることからわかる.

Pred 
$$(\mathbb{N}^2, \{(0,2)\}) \longrightarrow (\mathbb{N}, \{n \mid n \text{ is even}\})$$
 $p \downarrow$ 

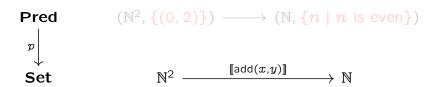
Set  $\mathbb{N}^2 \xrightarrow{[\![\mathsf{add}(x,y)]\!]} \mathbb{N}$ 

# ファイブレーションを使ったプログラムの解釈の具体例

変数  $x \ge y$  が x = 0 かつ y = 2 という条件を満たしているならば,プログラム add(x, y) を実行した結果の値は<mark>偶数</mark>である.

```
x : \text{Int}, y : \text{Int} \{x = 0 \land y = 2\}
 \vdash \text{add}(x, y) : \text{Int} \{v.v \text{ is even}\}
```

この主張が正しいことは,プログラム  $\operatorname{add}(x,y)$  の  $\operatorname{Set}$  における解釈がファイブレーション  $\operatorname{p}$  に沿って  $\operatorname{Pred}$  に持ち上がることからわかる.



# ファイブレーションを使ったプログラムの解釈の具体例

変数  $x \ge y$  が x = 0 かつ y = 2 という条件を満たしているならば,プログラム add(x, y) を実行した結果の値は<mark>偶数</mark>である.

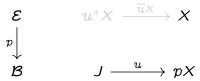
```
x: \text{Int}, y: \text{Int } \{x = 0 \land y = 2\}
 \vdash \text{add}(x, y): \text{Int } \{v.v \text{ is even}\}
```

この主張が正しいことは,プログラム  $\operatorname{add}(x,y)$  の  $\operatorname{Set}$  における解釈がファイブレーション  $\operatorname{p}$  に沿って  $\operatorname{Pred}$  に持ち上がることからわかる.

Pred 
$$(\mathbb{N}^2, \{(0,2)\}) \longrightarrow (\mathbb{N}, \{n \mid n \text{ is even}\})$$

$$\downarrow p$$
Set  $\mathbb{N}^2 \xrightarrow{[\![\mathsf{add}(x,y)]\!]} \mathbb{N}$ 

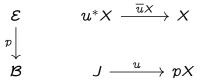
ファイブレーションの定義に ,  $u:J\to pX$  に対して  $\overline{u}X:u^*X\to X$  があってカルテシアン射となる , というもの があった .



プログラムのファイブレーション意味論の文脈では, $u^*X$  はプログラムの最弱事前条件に対応している.



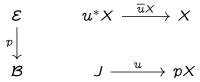
ファイブレーションの定義に ,  $u:J \to pX$  に対して  $\overline{u}X:u^*X \to X$  があってカルテシアン射となる , というもの があった .



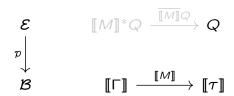
プログラムのファイブレーション意味論の文脈では ,  $u^*X$  はプログラムの最弱事前条件に対応している .



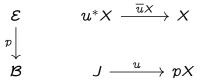
ファイブレーションの定義に ,  $u:J \to pX$  に対して  $\overline{u}X:u^*X \to X$  があってカルテシアン射となる , というもの があった .



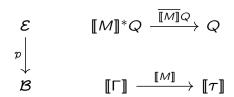
プログラムのファイブレーション意味論の文脈では, $u^*X$  はプログラムの最弱事前条件に対応している.



ファイブレーションの定義に , u:J o pX に対して  $\overline{u}X:u^*X o X$  があってカルテシアン射となる , というもの があった .



プログラムのファイブレーション意味論の文脈では, $u^*X$  はプログラムの最弱事前条件に対応している.

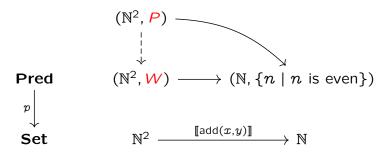


## 逆像関手の役割を具体例で観察

#### W に入る最も弱い条件は何か.

 $x: {
m Int}\, \{ {\it W} \} \vdash {
m add}(x,y): {
m Int}\, \{ v.v \ {
m is even} \}$ 「最も弱い」というのは

 $x: \operatorname{Int}, y: \operatorname{Int} \{ P \} \vdash \operatorname{add}(x,y): \operatorname{Int} \{ v.v \text{ is even} \}$  を満たす任意の条件 P に対して  $P \Rightarrow W$  が成り立つこと.



これは W が逆像の普遍性を持つということを言っている.

# 具体的な計算

```
x : Int, y : Int \{ \mathcal{W} \} \vdash add(x, y) : Int \{ v.v \text{ is even} \}
\{v.v \text{ is even}\} に対するプログラム add(x,y) の最弱事前条件
W \downarrow t .
            [add(x,y)]^*(\mathbb{N}, \{n \mid n \text{ is even}\})
             = (\mathbb{N}^2, [add(x, y)]^{-1} \{n \mid n \text{ is even}\})
             = (\mathbb{N}^2, \{(x, y) \mid x + y \text{ is even}\})
より「x + y is even」である.
```

## 具体的な計算

```
x : Int, y : Int \{ \mathcal{W} \} \vdash add(x, y) : Int \{ v.v \text{ is even} \}
\{v.v \text{ is even}\} に対するプログラム add(x,y) の最弱事前条件
W \downarrow t .
           [add(x,y)]^*(\mathbb{N}, \{n \mid n \text{ is even}\})
            = (\mathbb{N}^2, [add(x, y)]^{-1} \{n \mid n \text{ is even}\})
            = (\mathbb{N}^2, \{(x, y) \mid x + y \text{ is even}\})
より「x + y is even」である.
                x : Int, y : Int \{x = 0 \land y = 2\}
                 \vdash add(x,y): Int \{v,v\} is even
も成り立っていたが、このとき
               x = 0 \land y = 2 \Rightarrow x + y is even
が成り立っている.
```

# 最近考えていること