情報数学
第2回　原始帰納的関数

萩野　達也
hagino@sfc.keio.ac.jp

スライドURL

https://vu5.sfc.keio.ac.jp/slide/

前回

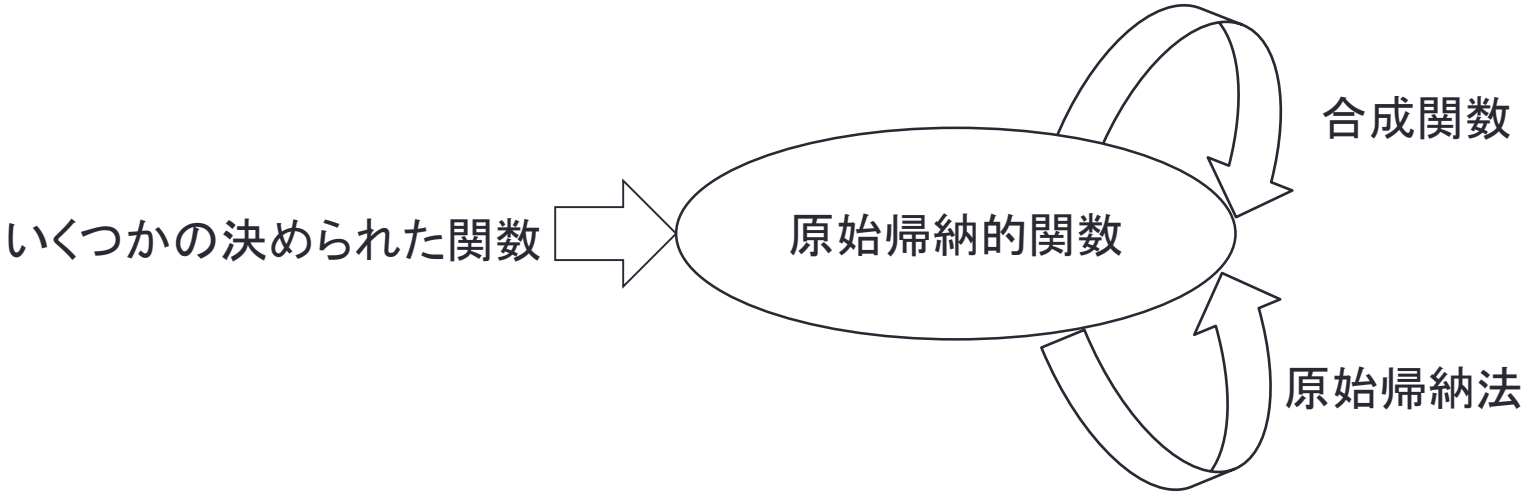
- 計算とは何か？
  - 計算 = コンピュータが計算しているもの
  - 計算可能関数 = コンピュータで計算している数学的な関数
  - 計算可能性 = どのような関数をコンピュータは計算できるのか
- whileプログラムとフローチャートは同じ計算能力がある
- どんなプログラムもwhileが一つだけのプログラムに書き換えることができる

原始帰納的関数

定義：

- 原始帰納的関数（primitive recursive function）は以下のものからなっている.

1. いくつかの決められた関数
2. 原始帰納的関数の合成関数
3. 原始帰納法によって定義された関数



いくつかの決められた原始帰納的関数

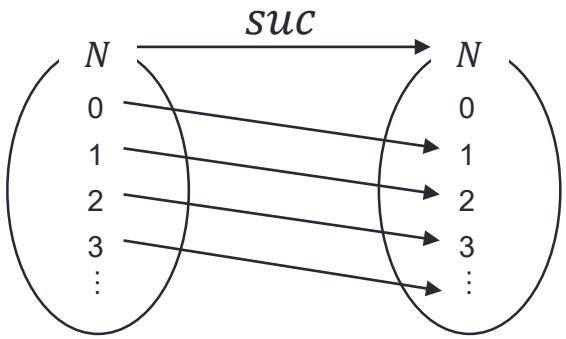
- 以下の3種類の関数は原始帰納的関数

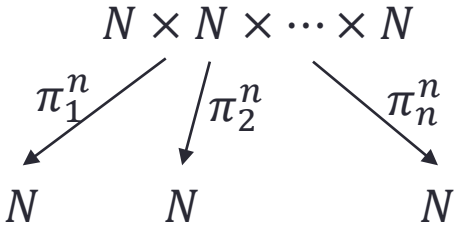
1. $zero: N^0 \to N$
   - $zero() = 0$
   - 常に 0 を返す関数

2. $suc: N \to N$
   - $suc(x) = x + 1$
   - x の次の数を返す関数
   - $suc(2) = 3$
   - $suc(100) = 101$

3. $\pi_i^n : N^n \to N$
   - $\pi_i^n(x_1, \ldots, x_n) = x_i$
   - n 個の引数の i 番目の要素を返す関数
   - 射影(projection)
   - $\pi_1^2(x, y) = x$
   - $\pi_2^2(x, y) = y$

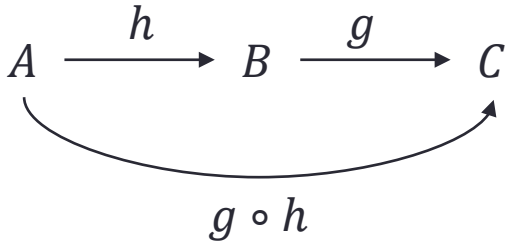
- $N^0 = \{\cdot\}$
- $N^1 = N$
- $N^2 = N \times N = \{(x, y) | x \in N, y \in N\}$
- $N^3 = N \times N \times N$
- $N^n = N \times N \times N \times \cdots \times N$



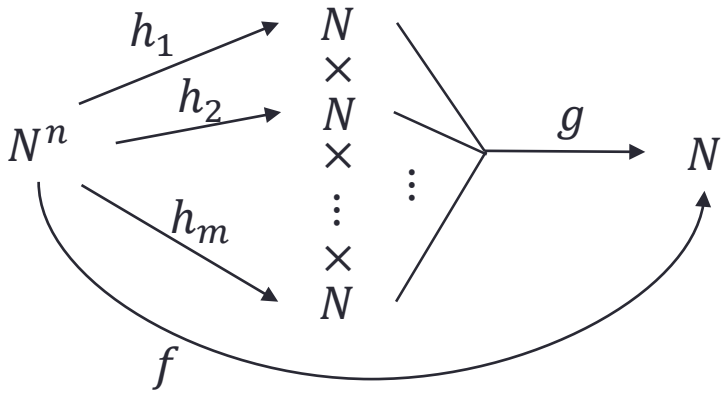


原始帰納関数の合成

- 関数の合成
  - 2つの関数を結合（合成）する
  - $g \circ h(x) = g(h(x))$



- 原始帰納的関数の合成は原始帰納的関数
  - $g : N^m \to N$ と $h_i : N^n \to N$ $(i = 1,2, \ldots, m)$ が原始帰納的関数のとき，合成した $f : N^n \to N$ も原始帰納的関数である．
  - $f(x_1, x_2, \ldots, x_n) = g(h_1(x_1, x_2, \ldots, x_n), \ldots, h_m(x_1, x_2, \ldots, x_n))$



例1:原始帰納的関数

- $id(x) = x$ $\quad id : N \to N$
  - 恒等関数は原始帰納的関数である.
  - $id(x) = \pi_1^1(x)$
- $one() = 1$ $\quad one : N^0 \to N$
  - 常に 1 を返す関数は原始帰納的関数である.
  - $one() = suc(zero())$
- $two() = 2$ $\quad two : N^0 \to N$
  - 定数を返す関数は原始帰納的関数である.
  - $two() = suc(suc(zero()))$
- $add2(x) = x + 2$ $\quad add2 : N \to N$
  - 与えられた数に2を加える関数は原始帰納的関数である.
  - $add2(x) = suc(suc(x))$

原始帰納法によって定義された関数

- $g : N^n \to N$ と $h : N^{n+2} \to N$ が原始帰納的関数のとき，次のように定義した $f : N^{n+1} \to N$ も原始帰納的関数
  - $f(x_1, \ldots, x_n, 0) = g(x_1, \ldots, x_n)$
  - $f\big(x_1, \ldots, x_n, suc(y)\big) = h(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y))$
- 原始帰納法による関数の定義と呼ばれる
- $n = 0$ の場合，$g : N^0 \to N$ と $h : N \times N \to N$ が原始帰納的関数であるとき，$f : N \to N$ は原始帰納的関数
  - $f(0) = g()$
  - $f\big(suc(y)\big) = h(y, f(y))$
- $f(x)$ を x が 0 である場合と，そうでない場合に分けて定義
  - x が 0 のときは $g()$ で定数
  - x が 0 でない場合は $x = suc(y)$ と書くことができるので，y と $f(y)$ から $f(x)$ の値を$h(y, f(y))$ で計算
  - $f(y)$ については，y が 0 であれば $g()$ であるが，そうでない場合にはもう一度 $f(suc(y))$ の場合を適用し，0 になるまでこれを繰り返す
- $f(1) = f\big(suc(0)\big) = h\big(0, f(0)\big) = h(0, g())$
- $f(2) = f\big(suc(1)\big) = h\big(1, f(1)\big) = h\big(1, h(0, g())\big)$
- $f(3) = f\big(suc(2)\big) = h\big(2, f(2)\big) = h\Big(2, h\big(1, h(0, g())\big)\Big)$

例2：原始帰納的関数

- $double: N \to N \qquad double(x) = x \times 2$
  - 数の2倍する関数を原始帰納法で定義する.
  - $double(0) = 0$
  - $double\big(suc(y)\big) = suc\big(suc\big(double(y)\big)\big)$

原始帰納法による定義$(n = 0)$
- $f(0) = g()$
- $f\big(suc(y)\big) = h(y, f(y))$

- $g() = zero()$
- $h(y, z) = \pi_1^2(y, suc\big(suc(z)\big))$

- $double(1) = double\big(suc(0)\big)$

$$= suc\Big(suc\big(double(0)\big)\Big) = suc\big(suc(0)\big) = 2$$

- $double(2) = double\Big(suc\big(suc(0)\big)\Big)$

$$= suc\Big(suc\Big(double\big(suc(0)\big)\Big)\Big) = suc\big(suc(2)\big) = 4$$

例3：原始帰納的関数

- $pred: N \to N$　　$pred(x) = x - 1$
  - 一つ前の数を返す関数(suc の逆)を原始帰納法で定義する.

  - $pred(0) =$

  - $pred\big(suc(y)\big) =$

  - $pred(1) =$

  - $pred(5) =$

原始帰納法による定義($n = 0$)
- $f(0) = g()$
- $f\big(suc(y)\big) = h(y, f(y))$

例4：原始帰納的関数

- $add: N^2 \to N$　　$add(x, y) = x + y$　　加算

  - $add(x, 0) = \ x$
  - $add\big(x, suc(y)\big) =$

原始帰納法による定義 $(n = 1)$
- $f(x, 0) = g(x)$
- $f\big(x, suc(y)\big) = h(x, y, f(x, y))$

  - $add(1,1) =$

  - $add(3,2) =$

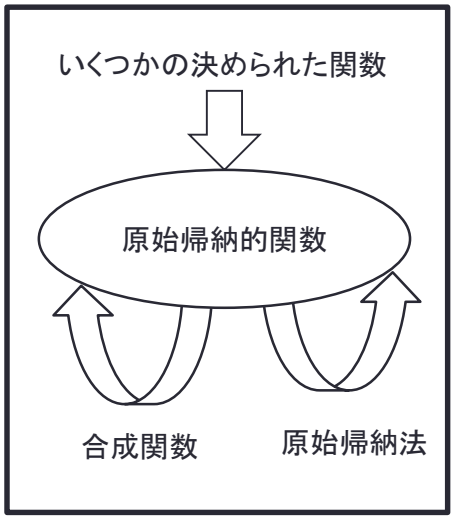
例5：原始帰納的関数

- $sub: N^2 \to N$　　$sub(x, y) = x - y$　減算
  - $sub(x, 0) = \quad x$
  - $sub\big(x, suc(y)\big) =$

原始帰納法による定義 $(n = 1)$
- $f(x, 0) = g(x)$
- $f\big(x, suc(y)\big) = h(x, y, f(x, y))$

- $mul: N^2 \to N$　　$mul(x, y) = x \times y$　乗算
  - $mul(x, 0) =$
  - $mul\big(x, suc(y)\big) =$

原始帰納的関数



- **Definition:**
  - 次の関数を原始帰納的関数（primitive recursive function
  - いくつかの決められた関数は原始帰納的関数である.
    - $zero: N^0 \to N \qquad zero() = 0$
    - $suc: N \to N \qquad suc(x) = x + 1 \quad (x\text{の次の数})$
    - $\pi_i^n : N^n \to N \qquad \pi_i^n(x_1, \dots, x_n) = x_i$

    簡単のために，今後 $zero()$ を 0 と書く
  - 原始帰納的関数の合成は原始帰納的関数である.
    - $g: N^m \to N$ と $h_i: N^n \to N$ $(i = 1,2, \dots, m)$ が原始帰納的関数のとき，合成した $f: N^n \to N$ も原始帰納的関数である.

      $$f(x_1, x_2, \dots, x_n) = g(h_1(x_1, x_2, \dots, x_n), \dots, h_m(x_1, x_2, \dots, x_n))$$
  - 原始帰納法による関数の定義
    - $g: N^n \to N$ と $h: N^{n+2} \to N$ が原始帰納的関数のとき，次のように定義した $f: N^{n+1} \to N$も原始帰納的関数である.

      $$f(x_1, \dots, x_n, 0) = g(x_1, \dots, x_n)$$
      $$f\big(x_1, \dots, x_n, suc(y)\big) = h(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y))$$

和と積

- **補題:**
  - $f(x_1, \ldots, x_n, y)$ が原始帰納的関数であれば，次の2つは原始帰納的関数である.

$$g(x_1, \ldots, x_n, y) = \sum_{z=0}^{y-1} f(x_1, \ldots, x_n, z)$$

$$h(x_1, \ldots, x_n, y) = \prod_{z=0}^{y-1} f(x_1, \ldots, x_n, z)$$

  - $n = 0$ の場合
    - $g(y) = f(0) + f(1) + \cdots + f(y-1)$
    - $h(y) = f(0) \times f(1) \times \cdots \times f(y-1)$
- **証明:**
  - $g(x_1, \ldots, x_n, 0) = 0$
  - $g\big(x_1, \ldots, x_n, suc(z)\big) = add\big(f(x_1, \ldots, x_n, z), g(x_1, \ldots, x_n, z)\big)$
  - $h(x_1, \ldots, x_n, 0) = 1$
  - $h\big(x_1, \ldots, x_n, suc(z)\big) = mul(f(x_1, \ldots, x_n, z), g(x_1, \ldots, x_n, z))$

原始帰納的述語

- 定義：
  - 述語 $P(x_1, \ldots, x_n): N^n \to \{T, F\}$ は，その特性関数 $C_P(x_1, \ldots, x_n): N^n \to N$ 原始帰納的関数であるとき，原始帰納的述語とよぶ．
    - $C_P(x_1, \ldots, x_n) = 1$　　（$P(x_1, \ldots, x_n)$ が T のとき）
    - $C_P(x_1, \ldots, x_n) = 0$　　（$P(x_1, \ldots, x_n)$ が F のとき）

- 例：
  - 述語「$x = 0$」は原始帰納的述語である．
    - $C_{=0}(x) = \quad 1 - x$
  - 述語「$x \leq y$」は原始帰納的述語である．
    - $C_{\leq}(x, y) = \ C_{=0}(x - y) = 1 - (x - y)$
  - 述語「$x = y$」は原始帰納的述語である．
    - $C_{=}(x, y) = \ C_{=0}\big((x - y) + (y - x)\big) = 1 - \big((x - y) + (y - x)\big)$

論理式

- **補題:**
  - $P(x_1, \ldots, x_n)$, $Q(x_1, \ldots, x_n)$, $R(x_1, \ldots, x_n, z)$ が原始帰納的述語であれば，次の述語も原始帰納的述語である.
    - $P(x_1, \ldots, x_n) \wedge Q(x_1, \ldots, x_n)$
    - $P(x_1, \ldots, x_n) \vee Q(x_1, \ldots, x_n)$
    - $\neg P(x_1, \ldots, x_n)$
    - $\forall z < y\big(R(x_1, \ldots, x_n, z)\big) \equiv \forall z\big(z < y \Rightarrow R(x_1, \ldots, x_n, z)\big)$
    - $\exists z < y\big(R(x_1, \ldots, x_n, z)\big) \equiv \exists z\big(z < y \wedge R(x_1, \ldots, x_n, z)\big)$

証明:

- $C_{P \wedge Q}(x_1, \ldots, x_n) = C_P(x_1, \ldots, x_n) \times C_Q(x_1, \ldots, x_n)$
- $C_{\neg P}(x_1, \ldots, x_n) = 1 - C_P(x_1, \ldots, x_n)$
- $C_{P \vee Q}(x_1, \ldots, x_n) = C_{\neg(\neg P \wedge \neg Q)}(x_1, \ldots x_n) = 1 - \big(1 - C_P(x_1, \ldots x_n)\big) \times \Big(1 - C_Q(x_1, \ldots, x_n)\Big)$
- $C_{\forall R}(x_1, \ldots, x_n, y) = \prod_{z=0}^{y-1} C_R(x_1, \ldots, x_n, z)$
- $C_{\exists R}(x_1, \ldots, x_n, y) = C_{\neg \forall \neg R}(x_1, \ldots, x_n, y) = 1 - \prod_{z=0}^{y-1}\big(1 - C_R(x_1, \ldots, x_n, z)\big)$

例

- 「$x > y$」は原始帰納的述語である.
  - $(x > y) \equiv$
- $divisible(x, y) \equiv$ 「x は y で割り切れる」は原始帰納的述語である.
  - $divisible(x, y) \equiv$
- $prime(x) \equiv$ 「x は素数である」は原始帰納的述語である.
  - $prime(x) \equiv$

最小値

- **補題**:
  - $P(x_1, \ldots, x_n, z)$ が原始帰納的述語であるとき，次の関数は原始帰納的関数である．

$$\mu_{z<y} P(x_1, \ldots, x_n, z) = \min(\{z \mid P(x_1, \ldots, x_n, z)\} \cup \{y\})$$

y 以下で $P(x_1, \ldots, x_n, z)$ が成り立つ最小の z を求める．

- **証明**:
  - $g(x_1, \ldots, x_n, z)$ を $\forall v < z(\neg P(x_1, \ldots, x_n, v))$ の特性関数とすると，

$$\mu_{z<y} P(x_1, \ldots, x_n, z) = \sum_{z=0}^{y-1} g(x_1, \ldots, x_n, z+1)$$

$n = 1$の場合
$\mu_{z<y} P(x, z)$

z	0	1	2	3	4	5	6	…	$y-1$	y
$P(x,z)$	F	F	F	T	F	T	T	…	F	T
$\neg P(x,z)$	T	T	T	F	T	F	F	…	T	F
$\forall v < z(\neg P(x,v))$	T	T	T	T	F	F	F		F	F
$g(x,z)$	1	1	1	1	0	0	0	…	0	0

$$\mu_{z<y} P(x,z) = \sum_{z=0}^{y-1} g(x, z+1) = \sum_{z=1}^{y} g(x,z) = 3$$

系

- **系:** $div(x,y) = x \div y$ は原始帰納的関数である.
  - $div(x,y) = \mu_{z<x}\left(x < \left(y \times (z+1)\right)\right)$
  - $div(5,2) =$

- **定理:** 四則演算(加算, 減算, 乗算, 除算)は原始帰納的関数である.

系: $pr(x) =$「x 番目の素数」は原始帰納的関数である.

- $pr(0) = 2$
- $pr\big(suc(x)\big) = \mu_{y<pr(x)!+2}\big(pr(x) < y \ \wedge \ prime(y)\big)$

まとめ

- 原始帰納的関数
  - $zero$
  - suc
  - π_i^n
  - 関数合成
  - 原始帰納法
- 原始帰納的述語
- 四則演算は原始帰納的関数である.