

「計算モデルの基礎理論」ノート  
井田哲雄著，岩波書店

田村直之 (神戸大学工学部)

# 1 計算モデルへの招待

## 1.1 計算モデルとは何か

- 1900年パリの国際数学会議で、ヒルベルトが示した23の問題の中の第10問題「不定方程式の有理整数解が存在するか否かを有限回的手段で判定すること」(不定方程式の例:  $x^3 + y^3 = z^3$ )
- 2未知数の場合: 肯定的解決 (A. Baker 1968). 一般の場合: 否定的解決 (Yu. V. Matiyasevich 1970).
- このような問題を決定問題と言う.

### メモ

- 国際数学会議ではフィールズ賞 (1936年より34名) とネヴァンリンナ賞 (計算機科学の数学的業績に与えられる) の表彰が行なわれる. 日本人のフィールズ賞授賞者は, 小平邦彦 (1954), 広中平祐 (1970), 森重文 (1990) の各氏である.
- フェルマーの最終定理は  $n = 1,000,000$  まで解がないことが証明されている.
- ところが, とうとう350年ぶりにフェルマーの最終定理が証明されたい! 証明したのは, プリンストン大学のアンドリュウ・ワイルズ博士 (40歳) である. 詳しくは数学セミナー 1993年9月号などを参照.

ヒルベルトの第10問題を契機として, 次のような問題が考えられた.

- 手段, アルゴリズム (algorithm) とは?
- 計算とは?
- 計算可能とは?

### 計算可能性の定義

数学基礎論の一分野として, 計算可能性 (computability) が研究されるようになった.

- Gödel, Kleene : 帰納的関数 (3章)
- Church, Kleene :  $\lambda$ 定義可能性 (4章)
- Turing : Turing 機械 (2.3節)

それぞれが, 等価であることが証明された.

**Church の提唱** 計算可能な関数とは帰納的関数のことである.

### 計算機の誕生

- 理論的計算可能性 → 実際の計算可能性
- 計算量理論 (theory of computational complexity)  
Turing 機械上で計算するのに必要な時間と領域の解析.  
 $P = NP?$  巡回セールスマン問題, ナップサック問題 (NP 完全).  
1990年ネヴァンリンナ賞 (ラズボロフ「クリーク問題 (頂点数  $k$  の完全部分グラフの存在) の単調計算量 (AND, OR のみ) は多項式的量を越える」).

- 現実の計算機上での計算量を解析するため、フォン・ノイマン・アーキテクチャに基づいた計算モデルの必要性が生じた。

Aho, Hopcroft, Ullman : RAM モデル (2.4 節)

### プログラミング方法論の発達

- ソフトウェア危機: プログラムの数学的取り扱いの必要性  
職人の世界 → 科学者の世界
- フォン・ノイマン・アーキテクチャの限界: 数学的取り扱いの困難さ。
- 新しい計算モデルに基づいたプログラミング言語
  - 関数モデル (3 章, 4 章)
  - 論理モデル (5 章)
  - 書換えモデル (6 章)
  - 代数モデル (7 章)

## 1.2 計算モデルの比較

## 2 機械モデル

### 2.1 基本的な考え方

### 2.2 有限状態機械

### 2.3 Turing 機械

### 2.4 RASP と RAM

### 3 関数モデル (1) 帰納的関数

#### 3.1 関数モデルの背景

#### 3.2 帰納的関数と計算の可能性

- 計算とは記号列の操作と考えることができる。
- 記号列の操作は、自然数上の計算に対応させることができる。  
実際、計算機上での処理は、メモリ全体を1つの2進数と考えて、それに対する計算とみなすことができる。
- 記号列に対応させる自然数を、その記号列の Gödel 数という。Gödel 数は、Gödel が不完全性定理を証明するために用いた。
- 自然数を  $0, 0', 0'', 0''', \dots$  で表す。また、これを  $0, 1, 2, 3, \dots$  で略記する。自然数の集合を  $\mathcal{N}$  で表す。

#### 3.3 初期関数

#### 3.4 合成

#### 3.5 原始帰納

#### 3.6 原始帰納関数 (原始帰納的関数)

定義 (原始帰納的関数, primitive recursive function)

原始帰納的関数を以下のように定義する。

(a) 次の関数は原始帰納的関数である (初期関数)。

- (1)  $z() = 0$  (零関数)
- (2)  $s(x) = x'$  (後者関数)
- (3)  $u_i^{(n)}(x_1, \dots, x_n) = x_i \quad (n, i \geq 1)$  (射影関数)

(b)  $g_1, \dots, g_m : \mathcal{N}^n \rightarrow \mathcal{N}, h : \mathcal{N}^m \rightarrow \mathcal{N} \quad (n, m \geq 0)$  が原始帰納的関数ならば、次のように定義される  $f : \mathcal{N}^n \rightarrow \mathcal{N}$  は原始帰納的関数である (合成)。

$$f(\vec{x}) = h(g_1(\vec{x}), \dots, g_m(\vec{x}))$$

ただし、 $\vec{x}$  は  $x_1, \dots, x_n$  の略記。  $m = 1$  のとき、 $f$  を  $h \circ g_1$  で表す。

(c)  $g : \mathcal{N}^n \rightarrow \mathcal{N}, h : \mathcal{N}^{n+2} \rightarrow \mathcal{N} \quad (n \geq 0)$  が原始帰納的関数ならば、次のように定義される  $f : \mathcal{N}^{n+1} \rightarrow \mathcal{N}$  は原始帰納的関数である (原始帰納法)。

$$\begin{aligned} f(0, \vec{y}) &= g(\vec{y}) \\ f(x', \vec{y}) &= h(x, f(x, \vec{y}), \vec{y}) \end{aligned}$$

ただし、 $\vec{y}$  は  $y_1, \dots, y_n$  の略記。  $f$  を  $\mathcal{P}r[g, h]$  で表す。

例 3.1  $n$  項の定数  $k$  関数  $c_k^{(n)} \quad (k \geq 0)$  は原始帰納的関数。

$$\begin{aligned} c_0^{(n)}(x_1, \dots, x_n) &= z() \\ c_1^{(n)} &= s \circ c_0^{(n)} \end{aligned}$$

$$\begin{aligned} c_2^{(n)} &= s \circ c_1^{(n)} \\ &\vdots \\ c_k^{(n)} &= s \circ c_{k-1}^{(n)} \end{aligned}$$

**例 3.2** 加算関数  $\text{plus}(x, y) = x + y$  は原始帰納的関数.

$$\begin{aligned} \text{plus}(0, y) &= y = u_1^{(1)}(y) \\ \text{plus}(x', y) &= s(\text{plus}(x, y)) = s(u_2^{(3)}(x, \text{plus}(x, y), y)) \\ &= s \circ u_2^{(3)}(x, \text{plus}(x, y), y) \end{aligned}$$

したがって,  $\text{plus} = \mathcal{P}r[u_1^{(1)}, s \circ u_2^{(3)}]$ .

**例 3.3** 1 を減じる関数 (前者関数)  $p(x)$  は原始帰納的関数.

$$\begin{aligned} p(0) &= 0 = z() \\ p(x') &= x = u_1^{(2)}(x, p(x)) \end{aligned}$$

したがって,  $p = \mathcal{P}r[z, u_1^{(2)}]$ .

**例 3.4** 減算関数  $\text{sub}(x, y) = x - y$  は原始帰納的関数.

$$\begin{aligned} \text{xsub}(0, x) &= x = u_1^{(1)}(x) \\ \text{xsub}(y', x) &= p(\text{xsub}(y, x)) = p(u_2^{(3)}(y, \text{xsub}(y, x), x)) \end{aligned}$$

したがって,  $\text{xsub} = \mathcal{P}r[u_1^{(1)}, p \circ u_2^{(3)}]$ .

$$\text{sub}(x, y) = \text{xsub}(u_2^{(2)}(x, y), u_1^{(2)}(x, y))$$

**例 3.5** 乗算関数  $\text{mult}(x, y) = x \times y$  は原始帰納的関数.

$$\begin{aligned} \text{mult}(0, y) &= 0 = c_0^{(1)}(y) \\ \text{mult}(x', y) &= \text{mult}(x, y) + y = \text{xplus}(x, \text{mult}(x, y), y) \\ \text{xplus}(x, y, z) &= \text{plus}(u_2^{(3)}(x, y, z), u_3^{(3)}(x, y, z)) \end{aligned}$$

したがって,  $\text{mult} = \mathcal{P}r[c_0^{(1)}, \text{xplus}]$ .

**例** 符号関数  $\text{sg}(x)$ ,  $\overline{\text{sg}}(x)$  を次のように定義する.

$$\begin{aligned} \text{sg}(x) &= \begin{cases} 0 & (x = 0) \\ 1 & (x > 0) \end{cases} \\ \overline{\text{sg}}(x) &= \begin{cases} 1 & (x = 0) \\ 0 & (x > 0) \end{cases} \end{aligned}$$

$\text{sg}(x)$ ,  $\overline{\text{sg}}(x)$  は原始帰納的関数.

$$\begin{aligned}\text{sg}(0) &= 0 = z() \\ \text{sg}(x') &= 0' = c_1^{(2)}(x, \text{sg}(x))\end{aligned}$$

したがって,  $\text{sg} = \mathcal{P}r[z, c_1^{(2)}]$ .

$$\begin{aligned}\overline{\text{sg}}(0) &= 0' = c_1^{(0)}() \\ \overline{\text{sg}}(x') &= 0 = c_0^{(2)}(x, \overline{\text{sg}}(x))\end{aligned}$$

したがって,  $\overline{\text{sg}} = \mathcal{P}r[c_1^{(0)}, c_0^{(2)}]$ .

**例**  $q, r, t: \mathcal{N}^n \rightarrow \mathcal{N}$  が原始帰納的関数のとき, 関数  $f: \mathcal{N}^n \rightarrow \mathcal{N}$  を次のように定義する.

$$f(\vec{x}) = \begin{cases} q(\vec{x}) & (t(\vec{x}) > 0) \\ r(\vec{x}) & (t(\vec{x}) = 0) \end{cases}$$

$f$  は原始帰納的関数である.

$$f(\vec{x}) = q(\vec{x}) \times \text{sg}(t(\vec{x})) + r(\vec{x}) \times \overline{\text{sg}}(t(\vec{x}))$$

この  $f(\vec{x})$  を **if**  $t(\vec{x})$  **then**  $q(\vec{x})$  **else**  $r(\vec{x})$  で表す.

**例** 剰余を計算する関数  $\text{rem}(x, y)$  は原始帰納的関数.

$$\begin{aligned}\text{rem}(0, y) &= 0 = z() \\ \text{rem}(x', y) &= \text{xrem}(x, \text{rem}(x, y), y) \\ \text{xrem}(x, r, y) &= \text{if } y \dot{-} s(r) \text{ then } s(r) \text{ else } 0\end{aligned}$$

したがって,  $\text{rem} = \mathcal{P}r[z, \text{xrem}]$ .

**例** 関数  $\text{div}(x, y)$  を次のように定義する.

$$\text{div}(x, y) = \begin{cases} 1 & (y|x) \\ 0 & (y \nmid x) \end{cases}$$

関数  $\text{div}$  は原始帰納的関数.

$$\text{div} = \overline{\text{sg}} \circ \text{rem}$$

**例** 商を求める関数  $\text{quo}(x, y)$  を次のように定義する.

$$\text{quo}(x, y) = \begin{cases} \lfloor \frac{x}{y} \rfloor & (y > 0) \\ 0 & (y = 0) \end{cases}$$

$\text{quo}(x, y)$  は原始帰納的関数.

$$\begin{aligned}\text{quo}(0, y) &= 0 \\ \text{quo}(x', y) &= \text{if } \text{div}(s(x), y) \text{ then } s(\text{quo}(x, y)) \text{ else } \text{quo}(x, y)\end{aligned}$$

例  $|x - y|$  は原始帰納的関数.

$$|x - y| = \text{if } x \dot{-} y \text{ then } x \dot{-} y \text{ else } y \dot{-} x$$

例  $p(z, \vec{y})$  が原始帰納的関数であり,  $x$  がある自然数のとき, 関数

$$\begin{aligned} \text{sum}(x, \vec{y}) &= \sum_{z=0}^x p(z, \vec{y}) \\ \text{prod}(x, \vec{y}) &= \prod_{z=0}^x p(z, \vec{y}) \end{aligned}$$

は原始帰納的関数である.

$$\begin{aligned} \text{sum}(0, \vec{y}) &= p(0, \vec{y}) \\ \text{sum}(x', \vec{y}) &= p(s(x), \vec{y}) + \text{sum}(x, \vec{y}) \\ \text{prod}(0, \vec{y}) &= p(0, \vec{y}) \\ \text{prod}(x', \vec{y}) &= p(s(x), \vec{y}) \times \text{prod}(x, \vec{y}) \end{aligned}$$

### 3.7 原始帰納述語 (原始帰納的述語)

原始帰納的関数のうち,  $\mathcal{N}^n \rightarrow \mathcal{B}$ ,  $\mathcal{B} = \{0, 1\}$  なるものを原始帰納的述語という. 0 を偽, 1 を真とみなす.

例 述語  $\text{eq}(x, y) = (x = y)$  を次のように定義する.

$$\text{eq}(x, y) = \begin{cases} 1 & (x = y) \\ 0 & (x \neq y) \end{cases}$$

$\text{eq}$  は原始帰納的述語.

$$\text{eq}(x, y) = \overline{\text{sg}}(|x - y|)$$

例 述語  $x > y$  は原始帰納的述語.

$$x > y = \text{sg}(x \dot{-} y)$$

#### 命題

述語  $p(\vec{x})$ ,  $q(\vec{y})$  が原始帰納的述語ならば,  $\neg p(\vec{x})$ ,  $p(\vec{x}) \wedge q(\vec{y})$ ,  $p(\vec{x}) \vee q(\vec{y})$  で表される述語も原始帰納的述語である.

$$\begin{aligned} \neg p(\vec{x}) &= \overline{\text{sg}}(p(\vec{x})) \\ p(\vec{x}) \wedge q(\vec{y}) &= p(\vec{x}) \times q(\vec{y}) \\ p(\vec{x}) \vee q(\vec{y}) &= \neg(\neg p(\vec{x}) \wedge \neg q(\vec{y})) \end{aligned}$$

**命題 (有界限量)**

述語  $p(z, \vec{y})$  が原始帰納的述語であり,  $x$  がある自然数のとき,  $(\forall z)_{\leq x} p(z, \vec{y})$  あるいは  $(\exists z)_{\leq x} p(z, \vec{y})$  で表される述語も原始帰納的述語である.

$$\begin{aligned} (\forall z)_{\leq x} p(z, \vec{y}) &= \prod_{z=0}^x p(z, \vec{y}) \\ (\exists z)_{\leq x} p(z, \vec{y}) &= \neg(\forall z)_{\leq x} \neg p(z, \vec{y}) \end{aligned}$$

**例**  $x$  が素数のとき真となる述語  $\text{prime}(x)$  は原始帰納的述語.

$$\text{prime}(x) = (x > 0') \wedge (\forall z)_{\leq x} ((z = 0') \vee (z = x) \vee \neg \text{div}(x, z))$$

**命題 (有界最小化)**

述語  $p : \mathcal{N}^{n+1} \rightarrow \mathcal{B}$  が原始帰納的述語で,  $z$  がある自然数のとき, 関数  $f : \mathcal{N}^{n+1} \rightarrow \mathcal{N}$  を次のように定義する.

$$f(z, \vec{x}) = \begin{cases} y & (y \leq z \text{ が } p(y, \vec{x}) = 1 \text{ となる最小の値のとき}) \\ z + 1 & (y \leq z \text{ で } p(y, \vec{x}) = 1 \text{ となる } y \text{ が存在しないとき}) \end{cases}$$

$f$  は原始帰納的関数である.

$$f(z, \vec{x}) = \sum_{i=0}^z \prod_{y=0}^i p(y, \vec{x})$$

$f(z, \vec{x})$  を  $(\mu y)_{\leq z} [p(y, \vec{x})]$  で表す.

**3.8 最小化****定義 (帰納的関数)**

帰納的関数を以下のように定義する.

(a)–(c) 原始帰納的関数の定義で「原始帰納的関数」を「帰納的関数」で置き換えた文.

(d)  $p : \mathcal{N}^{n+1} \rightarrow \mathcal{B}$  が帰納的関数で,  $\forall \vec{x} \exists y; p(y, \vec{x}) = 1$  のとき, 次のように定義される  $f : \mathcal{N}^{n+1} \rightarrow \mathcal{N}$  は帰納的関数である (最小化).

$$f(\vec{x}) = y \quad (y \text{ は } p(y, \vec{x}) = 1 \text{ となる最小の値})$$

$f(\vec{x})$  を  $(\mu y)[p(y, \vec{x})]$  で表す.

注意: 帰納的関数は全域関数である.

**定義 (帰納的部分関数)**

帰納的部分関数を以下のように定義する.

(a)–(c) 原始帰納的関数の定義で「原始帰納的関数」を「帰納的部分関数」で置き換えた文.

(d)  $p : \mathcal{N}^{n+1} \rightarrow \mathcal{B}$  が帰納的部分関数のとき、次のように定義される  $f : \mathcal{N}^{n+1} \rightarrow \mathcal{N}$  は帰納的部分関数である (最小化).

$$f(\vec{x}) = \begin{cases} y & (y \text{ が } p(y, \vec{x}) = 1 \text{ となる最小の値のとき}) \\ \text{未定義} & (p(y, \vec{x}) = 1 \text{ となる } y \text{ が存在しないとき}) \end{cases}$$

$f(\vec{x})$  を  $(\mu y)[p(y, \vec{x})]$  で表す.

### 3.9 対の自然数へのコーディング法

#### 定義

関数  $\phi : \mathcal{N} \times \mathcal{N} \rightarrow \mathcal{N}$ ,  $\text{hd} : \mathcal{N} \rightarrow \mathcal{N}$ ,  $\text{tl} : \mathcal{N} \rightarrow \mathcal{N}$  を次のように定義する.

$$\begin{aligned} \phi(x, y) &= \frac{1}{2}(x+y)(x+y+1) + y + 1 \\ \text{hd}(z) &= f((\mu x)_{\leq z}[(\exists y)_{\leq z}(\phi(x, y) = z)], z) \\ \text{tl}(z) &= f((\mu y)_{\leq z}[(\exists x)_{\leq z}(\phi(x, y) = z)], z) \\ f(w, z) &= \text{if } w \dot{=} z \text{ then } 0 \text{ else } w \end{aligned}$$

$\phi(x, y)$  を  $\langle x, y \rangle$  で表す.

#### 命題

- $\phi$ ,  $\text{hd}$ ,  $\text{tl}$  は原始帰納的関数である.
- $\phi$  は  $\mathcal{N} \times \mathcal{N}$  と正の整数の間の 1 対 1 対応である.
- $\text{hd}(\langle x, y \rangle) = x$
- $\text{tl}(\langle x, y \rangle) = y$

#### 定義 (リスト)

リスト  $[x_1, \dots, x_n]$  ( $n \geq 0$ ) を次のように定義する.

$$\begin{aligned} [] &= 0 \\ [x_1, \dots, x_n] &= \langle x_1, \dots, \langle x_n, [] \rangle \dots \rangle \quad (n > 0) \end{aligned}$$

関数  $\pi_i$  ( $i > 0$ ) を次のように定義する.

$$\begin{aligned} \pi_i &= \text{hd} \circ \text{tl}^{i-1} \\ \pi(i, x) &= \pi_i(x) \end{aligned}$$

#### 命題

- $\text{hd}([x_1, \dots, x_n]) = x_1 \quad (n > 0)$
- $\text{tl}([x_1, \dots, x_n]) = [x_2, \dots, x_n] \quad (n > 0)$
- $\pi_i([x_1, \dots, x_n]) = x_i \quad (0 < i \leq n)$

- $\pi(i, [x_1, \dots, x_n]) = x_i \quad (0 < i \leq n)$
- $\pi_i, \pi$  は原始帰納的関数.

$$\begin{aligned}\pi(i, x) &= \text{hd}(f(i-1, x)) \\ f(0, x) &= x \\ f(i', x) &= \text{tl}(f(i, x))\end{aligned}$$

### 命題 (同時原始帰納法)

関数  $g_i, h_i \ (1 \leq i \leq n)$  が原始帰納的関数ならば, 次のように定義される関数  $f_i \ (1 \leq i \leq n)$  は原始帰納的関数である.

$$\begin{aligned}f_i(0, \vec{y}) &= g_i(\vec{y}) \\ f_i(x', \vec{y}) &= h_i(x, \vec{y}, f_1(x, \vec{y}), \dots, f_n(x, \vec{y}))\end{aligned}$$

**証明** 次のように関数  $\vec{f}, \vec{g}, \vec{h}$  をおく.

$$\begin{aligned}\vec{f}(x, \vec{y}) &= [f_1(x, \vec{y}), \dots, f_n(x, \vec{y})] \\ \vec{g}(\vec{y}) &= [g_1(\vec{y}), \dots, g_n(\vec{y})] \\ \vec{h}(x, z, \vec{y}) &= [h_1(x, \vec{y}, \pi_1(z), \dots, \pi_n(z)), \dots, h_n(x, \vec{y}, \pi_1(z), \dots, \pi_n(z))]\end{aligned}$$

このとき,  $\vec{f}$  は次のように表せる.

$$\begin{aligned}\vec{f}(0, \vec{y}) &= \vec{g}(\vec{y}) \\ \vec{f}(x', \vec{y}) &= \vec{h}(x, \vec{f}(x, \vec{y}), \vec{y})\end{aligned}$$

したがって  $\vec{f}$  は原始帰納的関数である.  $f_i = \pi_i \circ \vec{f}$  であるから,  $f_i$  も原始帰納的関数である.

**例** 偶数, 奇数を判定する述語 even, odd.

### 命題 (累積帰納法)

関数  $g, h$  が原始帰納的関数ならば, 次のように定義される関数  $f$  も原始帰納的関数である.

$$\begin{aligned}f(0, \vec{y}) &= g(\vec{y}) \\ f(x', \vec{y}) &= h(x, H(x, \vec{y}), \vec{y}) \\ H(x, \vec{y}) &= [f(x, \vec{y}), f(x-1, \vec{y}), \dots, f(0, \vec{y})]\end{aligned}$$

**証明**

$$\begin{aligned}H(0, \vec{y}) &= [f(0, \vec{y})] \\ H(x', \vec{y}) &= [f(x', \vec{y}), \dots, f(0, \vec{y})] \\ &= G(x, H(x, \vec{y}), \vec{y}) \\ G(x, z, \vec{y}) &= \langle h(x, z, \vec{y}), z \rangle\end{aligned}$$

であるから,  $H$  は原始帰納的関数である.  $f = \text{hd} \circ H$  であるから,  $f$  も原始帰納的関数である.

例 次のように定義される関数 fib.

$$\begin{aligned} \text{fib}(0) &= 1 \\ \text{fib}(1) &= 1 \\ \text{fib}(n) &= \text{fib}(n-1) + \text{fib}(n-2) \quad (n \geq 2) \end{aligned}$$

### 命題

関数  $r_1, \dots, r_m$  が原始帰納的関数であり,  $(\forall n > 0) r_i(n) < n$  ならば, 次のように定義される関数  $f$  も原始帰納的関数である.

$$\begin{aligned} f(0, \vec{y}) &= g(\vec{y}) \\ f(x', \vec{y}) &= h(x, f(r_1(x'), \vec{y}), \dots, f(r_m(x'), \vec{y}), \vec{y}) \end{aligned}$$

### 証明

$$h'(x, z, \vec{y}) = h(x, \pi(x+1-r_1(x'), z), \dots, \pi(x+1-r_m(x'), z), \vec{y})$$

とおく.  $z = [f(x, \vec{y}), f(x-1, \vec{y}), \dots, f(0, \vec{y})]$  のとき,  $\pi(i, z) = f(x+1-i, \vec{y})$  より,  $\pi(x+1-r(x'), z) = f(r(x'), \vec{y})$  である. したがって,  $f$  は累積帰納法を用いて

$$\begin{aligned} f(0, \vec{y}) &= g(\vec{y}) \\ f(x', \vec{y}) &= h'(x, H(x, \vec{y}), \vec{y}) \\ H(x, \vec{y}) &= [f(x, \vec{y}), f(x-1, \vec{y}), \dots, f(0, \vec{y})] \end{aligned}$$

と表せるから,  $f$  は原始帰納的関数である.

例 リスト  $x$  の長さを求める関数  $\text{len}(x)$  は原始帰納的関数である.  $\text{tl}(x') < x'$  に注意.

$$\begin{aligned} \text{len}(0) &= 0 \\ \text{len}(x') &= \text{s} \circ \text{u}_2^{(2)}(x, \text{len}(\text{tl}(x'))) \\ &= \text{s}(\text{len}(\text{tl}(x'))) \end{aligned}$$

## 3.10 帰納的関数のクラスと Church の提唱

### 定義 (原始帰納的関数のゲーデル数)

原始帰納的関数  $f: \mathcal{N}^n \rightarrow \mathcal{N}$  のゲーデル数  $\langle\langle f \rangle\rangle$  を以下のように定義する.

- 零関数  $z$  のとき  
 $\langle\langle z \rangle\rangle = \langle 0, 1 \rangle$
- 後者関数  $s$  のとき  
 $\langle\langle s \rangle\rangle = \langle 1, 2 \rangle$

- 射影関数  $u_i^{(n)}$  のとき  
 $\langle\langle u_i^{(n)} \rangle\rangle = \langle n, 3 + 3 \cdot i \rangle$
- 合成  $h(g_1(\vec{x}), \dots, g_m(\vec{x}))$  のとき  
 $\langle\langle f \rangle\rangle = \langle n, 4 + 3 \cdot [\langle\langle h \rangle\rangle, \langle\langle g_1 \rangle\rangle, \langle\langle g_2 \rangle\rangle, \dots, \langle\langle g_m \rangle\rangle] \rangle$
- 原始帰納  $f(0, \vec{y}) = g(\vec{y})$ ,  $f(x', \vec{y}) = h(x, f(x, \vec{y}), \vec{y})$  のとき  
 $\langle\langle f \rangle\rangle = \langle n, 5 + 3 \cdot [\langle\langle g \rangle\rangle, \langle\langle h \rangle\rangle] \rangle$

関数  $\langle\langle f \rangle\rangle$  は単射である.

### 命題

自然数  $x$  がある原始帰納的関数のゲーデル数となっているかどうかを判定する述語  $\text{GN}(x)$  は原始帰納的述語である.

$$\text{GN}(x) = \begin{cases} 1 & (\exists f; \langle\langle f \rangle\rangle = x) \\ 0 & (\text{その他}) \end{cases}$$

**証明** 関数  $G$  を次のように与える.

$$G([x_1, \dots, x_n]) = \prod_{i=1}^n \text{GN}(x_i)$$

$G(x)$  は次のように原始帰納的関数として定義できる.

$$\begin{aligned} G(0) &= 1 \\ G(x') &= G(\text{tl}(x')) \wedge ( \\ &\quad p_1(\text{hd}(\text{hd}(x')), \text{tl}(\text{hd}(x'))) \vee \\ &\quad p_2(\text{hd}(\text{hd}(x')), \text{tl}(\text{hd}(x'))) \vee \\ &\quad p_3(\text{hd}(\text{hd}(x')), \text{tl}(\text{hd}(x'))) \vee \\ &\quad (p_4(\text{hd}(\text{hd}(x')), \text{tl}(\text{hd}(x'))) \wedge G(\text{quo}(\text{tl}(\text{hd}(x')) \dot{-} 4, 3)) \vee \\ &\quad (p_5(\text{hd}(\text{hd}(x')), \text{tl}(\text{hd}(x'))) \wedge G(\text{quo}(\text{tl}(\text{hd}(x')) \dot{-} 5, 3)) ) \\ p_1(n, x) &= (n = 0) \wedge (x = 1) \\ p_2(n, x) &= (n = 1) \wedge (x = 2) \\ p_3(n, x) &= (x > 3) \wedge (\text{rem}(x, 3) = 0) \wedge (\text{quo}(x \dot{-} 3, 3) \leq n) \\ p_4(n, x) &= (x > 3) \wedge (\text{rem}(x, 3) = 1) \wedge p'_4(n, \text{quo}(x \dot{-} 4, 3)) \\ p_5(n, x) &= (x > 3) \wedge (\text{rem}(x, 3) = 2) \wedge p'_5(n, \text{quo}(x \dot{-} 5, 3)) \\ p'_4(n, x) &= (\text{len}(x) > 0) \wedge (\text{hd}(\text{hd}(x)) = \text{len}(x) \dot{-} 1) \wedge p''_4(\text{tl}(x), n) \\ p''_4(0, n) &= 1 \\ p''_4(x', n) &= (\text{hd}(\text{hd}(x')) = n) \wedge p''_4(\text{tl}(x'), n) \\ p'_5(n, x) &= (\text{len}(x) = 2) \wedge (\text{hd}(\pi_1(x)) = n \dot{-} 1) \wedge (\text{hd}(\pi_2(x)) = n + 1) \end{aligned}$$

$\text{GN}(x) = G([x])$  であるから, 述語  $\text{GN}$  は原始帰納的述語である.

**定義 (原始帰納的関数の計算)**

$n$  変数の原始帰納的関数  $f$  の  $[x_1, \dots, x_n]$  に対する計算を以下のように定義される自然数とする. 以下では,  $\vec{x} = x_1, \dots, x_n, \vec{y} = x_2, \dots, x_n$  とする.

- 零関数  $z$  のとき

$$[0, [], []]$$

- 後者関数  $s$  のとき

$$[x_1 + 1, [x_1], []]$$

- 射影関数  $u_i^{(n)}$  のとき

$$[x_i, [\vec{x}], []]$$

- 合成  $h(g_1(\vec{x}), \dots, g_m(\vec{x}))$  のとき

$$[\text{hd}(u), [\vec{x}], [u, v_1, \dots, v_m]]$$

ただし,  $v_i$  は  $g_i$  の  $[\vec{x}]$  に対する計算,  $u$  は  $h$  の  $[g_1(\vec{x}), \dots, g_m(\vec{x})]$  に対する計算である.

- 原始帰納  $f(0, \vec{y}) = g(\vec{y}), f(x', \vec{y}) = h(x, f(x, \vec{y}), \vec{y})$  のとき

$$[\text{hd}(u), [\vec{x}], [u]] \quad (x_1 = 0 \text{ のとき})$$

$$[\text{hd}(v), [\vec{x}], [v, w]] \quad (x_1 > 0 \text{ のとき})$$

ただし,  $u$  は  $g$  の  $[\vec{y}]$  に対する計算,  $w$  は  $f$  の  $[x_1 - 1, \vec{y}]$  に対する計算,  $v$  は  $h$  の  $[x_1 - 1, f(x_1 - 1, \vec{y}), \vec{y}]$  に対する計算である.

**命題**

与えられた自然数  $a, x, c$  について,  $c$  がゲーデル数  $a$  で表される原始帰納的関数の  $x$  に対する計算かどうかを判定する述語  $C(a, x, c)$  は原始帰納的関数である.

$$C(a, x, c) = \begin{cases} 1 & (c \text{ がゲーデル数 } a \text{ で表される原始帰納的関数の } x \text{ に対する計算の時}) \\ 0 & (\text{その他}) \end{cases}$$

**証明**  $c$  がゲーデル数  $a$  で表される原始帰納的関数の  $x$  に対する計算のとき, 定義より  $\pi_2(c) = x$  である. そこで, 関数  $G$  を次のように与える.

$$G([[a_1, c_1], \dots, [a_n, c_n]]) = \prod_{i=1}^n C(a_i, \pi_2(c_i), c_i)$$

$G(x)$  は原始帰納的関数として定義できる. 定義の詳細は省略する.  $f(\vec{x})$  が  $g(\vec{y})$  を用いて求められるとき,  $f$  の  $[\vec{x}]$  に対する計算をリスト  $c$ ,  $g$  の  $[\vec{y}]$  に対する計算をリスト  $d$  とすると,  $[\langle\langle f \rangle\rangle, c] > [\langle\langle g \rangle\rangle, d]$  となることがポイントとなる.

$$C(a, x, c) = (x = \pi_2(c)) \wedge G([[a, c]])$$

であるから, 述語  $C$  は原始帰納的述語である.

**定義 (万能原始帰納的関数)**

関数  $U : \mathcal{N}^2 \rightarrow \mathcal{N}$  を次のように定義する.

$$U(a, [x_1, \dots, x_n]) = \begin{cases} f(x_1, \dots, x_n) & (a \text{ がある } n \text{ 変数原始帰納的関数 } f \text{ のゲーデル数のとき}) \\ 0 & \text{その他のとき} \end{cases}$$

関数  $U$  は全域関数であり, 次のように帰納的関数として定義することができる.

$$U(a, x) = \text{if GN}(a) \wedge (\text{hd}(a) = \text{len}(x)) \text{ then hd}((\mu c)[C(a, x, c)]) \text{ else } 0$$

**命題**

原始帰納的関数, 帰納的関数は全域関数である.

**命題**

(全域) 帰納的関数のうち原始帰納的関数でないものが存在する.

**証明** 関数  $g : \mathcal{N} \rightarrow \mathcal{N}$  を次のように定義する.

$$g(x) = U(x, [x]) + 1$$

$g$  は 1 変数の帰納的関数であるが, すべての 1 変数原始帰納的関数と異なる.

$$\begin{aligned} g(\langle\langle f \rangle\rangle) &= U(\langle\langle f \rangle\rangle, [\langle\langle f \rangle\rangle]) + 1 \\ &= f(\langle\langle f \rangle\rangle) + 1 \end{aligned}$$

したがって, 帰納的関数のうち原始帰納的関数でないものが存在する (対角線論法).

**例 (アッカーマン関数)** 次のアッカーマン関数  $A(x, y)$  は, (全域) 帰納的関数のうち原始帰納的関数でない関数の例である (2 重数学的帰納法).

$$\begin{aligned} A(0, y) &= y + 1 \\ A(x + 1, 0) &= A(x, 1) \\ A(x + 1, y + 1) &= A(x, A(x + 1, y)) \end{aligned}$$

証明は, 小林孝次郎「計算可能性入門」近代科学社, 9.3 節などを参照.

**命題**

原始帰納的関数  $\subset$  帰納的関数  $\subset$  帰納的部分関数

**チャーチの提唱**

帰納的関数 = 計算可能な全域関数 = アルゴリズム (必ず停止) が存在

拡張された形:

帰納的部分関数 = 計算可能な部分関数 = 手続き (必ずしも停止しない) が存在

## 4 関数モデル (2) ラムダ計算

### 4.1 歴史的背景

### 4.2 基本的な考え方

### 4.3 ラムダ計算の言語

#### 定義 (λ項)

$\mathcal{V}$  を可算無限個の変数からなる集合とする.  $\lambda$ 項の集合 $\Lambda$ を以下のように定義する.

- (1)  $x \in \mathcal{V}$  のとき,  $x \in \Lambda$
- (2)  $M, N \in \Lambda$  のとき,  $(MN) \in \Lambda$  (作用表現)
- (3)  $M \in \Lambda, x \in \mathcal{V}$  のとき,  $(\lambda x.M) \in \Lambda$  (抽象化表現)
- (4) (1)(2)(3) 以外に $\Lambda$ の要素はない

$\lambda$ 項  $L, M$  が一致することを  $L \equiv M$  で表す.

#### 略記規則

- $M_1 M_2 \cdots M_n = ((\cdots (M_1 M_2) \cdots) M_n)$
- $\lambda x_1 \cdots x_n.M = (\lambda x_1. \cdots (\lambda x_n.M) \cdots)$
- $\lambda$ 項の最外側のカッコは省略してよい

#### 定義 (部分項)

$\lambda$ 項  $L$  の部分項を以下のように定義する.

- (1)  $L \in \mathcal{V}$  のとき,  $L$  は  $L$  の部分項
- (2)  $L \equiv (MN)$  のとき,  $M, N, L$  は  $L$  の部分項
- (3)  $L \equiv (\lambda x.M)$  のとき,  $M, L$  は  $L$  の部分項
- (4) (1)(2)(3) 以外に  $L$  の部分項はない

$L$  の部分項のうち  $L$  以外のものを  $L$  の真部分項と呼ぶ.

#### 定義 (正整数の列)

正整数の集合を  $\mathcal{N}_+$  で表す.  $i_1, \dots, i_n$  を  $n \geq 0$  個の  $\mathcal{N}_+$  の要素とするととき, その列を

$$i_1 \cdot i_2 \cdots i_n$$

で表す. 特に空列 ( $n = 0$ ) のときは  $\epsilon$  で表す. また, 列全体の集合を  $\mathcal{N}_+^*$  で表す.

$$\mathcal{N}_+^* = \{i_1 \cdot i_2 \cdots i_n \mid i_k \in \mathcal{N}_+, n \geq 0\}$$

#### 定義 (出現位置)

$\lambda$ 項  $M$  に対し, 出現位置の集合  $\mathcal{O}(M) \subset \mathcal{N}_+^*$  を次のように定義する.

- (1)  $M \equiv x \in \mathcal{V}$  のとき,  $\mathcal{O}(M) = \{\epsilon\}$
- (2)  $M \equiv (M_1 M_2)$  のとき,  $\mathcal{O}(M) = \{\epsilon\} \cup \{i \cdot u \mid u \in \mathcal{O}(M_i), i = 1, 2\}$
- (3)  $M \equiv (\lambda x.M_1)$  のとき,  $\mathcal{O}(M) = \{\epsilon\} \cup \{1 \cdot u \mid u \in \mathcal{O}(M_1)\}$

**定義 (木領域)**

次の性質を満たす  $\mathcal{N}_+^*$  の部分集合  $D$  を木領域という.

- (1)  $v \cdot w \in D, v, w \in \mathcal{N}_+^* \implies v \in D$
- (2)  $u \cdot i \in D, u \in \mathcal{N}_+^*, i \in \mathcal{N}_+ \implies u \cdot j \in D \quad (j = 1, \dots, i)$

**定義 (木領域上の関係  $\preceq, \prec$ )**

木領域  $D$  の関係  $\preceq, \prec$  を次のように定める.

- $u \preceq v \iff \exists w \in D, v = u \cdot w$
- $u \prec v \iff u \preceq v$  かつ  $u \neq v$

$\preceq$  は  $D$  上の半順序関係 (反射的, 推移的, 反対称的) である.

**定義 (自由変数, 束縛変数)**

- (1)  $M \in \mathcal{V}$  のとき,  $M/e$  は  $M$  において自由変数
- (2)  $M \equiv (M_1 M_2)$ ,  $M_i/u$  が  $M_i$  において自由 (束縛) 変数のとき,  $M/i \cdot u$  は  $M$  において自由 (束縛) 変数
- (3)  $M \equiv (\lambda x.M_1)$ ,  $M_1/u$  が  $M_1$  において
  - 自由変数かつ  $x \neq M_1/u$  のとき,  $M/1 \cdot u$  は  $M$  において自由変数
  - 自由変数かつ  $x = M_1/u$  のとき,  $M/1 \cdot u$  は  $M$  において束縛変数
  - 束縛変数のとき,  $M/1 \cdot u$  は  $M$  において束縛変数

$M$  中の自由変数の集合を  $\mathcal{V}(M)$  で表す.

**定義 (renaming)**

$\lambda$  項  $M$  において, 変数  $x$  から変数  $z$  への renaming は, 次のように定義される項  $\{x/z\}M$  である.

- (1)  $\{x/z\}x \equiv z$
- (2)  $\{x/z\}y \equiv y \quad (x \neq y \text{ のとき})$
- (3)  $\{x/z\}(M_1 M_2) \equiv (\{x/z\}M_1 \{x/z\}M_2)$
- (4)  $\{x/z\}(\lambda x.M_1) \equiv (\lambda z.\{x/z\}M_1)$
- (5)  $\{x/z\}(\lambda y.M_1) \equiv (\lambda y.\{x/z\}M_1) \quad (x \neq y \text{ のとき})$

**定義 ( $\alpha$ 変換)**

$\lambda$  項  $M$  中において,  $z$  が  $M$  に自由変数としても束縛変数としても現れないとき,  $(\lambda x.M)$  から  $(\lambda z.\{x/z\}M)$  への変換を  $\alpha$ 変換と呼び, 次のように表わす.

$$(\lambda x.M) \rightarrow_\alpha (\lambda z.\{x/z\}M)$$

**定義 ( $\alpha$ 合同)**

$\lambda$  項  $M$  と  $N$  について, 次のいずれかが成り立つとき  $M$  と  $N$  は  $\alpha$ 合同であるといい,  $M =_\alpha N$  で表す.

- (1)  $M \equiv N$
- (2)  $M \rightarrow_\alpha N$
- (3)  $S \rightarrow_\alpha T$  であり,  $M$  の部分項  $S$  を  $T$  で置き換えたものが  $N$
- (4) ある  $\lambda$  項  $R$  が存在し,  $M =_\alpha R$  かつ  $N =_\alpha R$

$\alpha$ 合同は同値関係である (反射的, 対称的, 推移的).

**定義 (代入)**

$M, N \in \Lambda$ ,  $x \in \mathcal{V}$  のとき, 代入  $M[x := N]$  は次のように定義される  $\lambda$  項である.

- (1)  $M \in \mathcal{V}$  のとき,
  - (1a)  $M = x$  のとき,  $M[x := N] \equiv N$
  - (1b)  $M \neq x$  のとき,  $M[x := N] \equiv M$
- (2)  $M \equiv (M_1 M_2)$  のとき,  $M[x := N] =_{\alpha} (M_1[x := N] M_2[x := N])$
- (3)  $M \equiv (\lambda y. M_1)$  のとき,  $M \rightarrow_{\alpha} (\lambda z. M_2)$  かつ  $x \neq z$  かつ  $z \notin \mathcal{V}(N)$  として,  $M[x := N] \equiv (\lambda z. M_2[x := N])$

**定義 (出現位置への代入)**

$M, N \in \Lambda$ ,  $u \in \mathcal{O}(M)$  のとき,  $M$  の出現位置  $u$  にある  $\lambda$  項を  $N$  で置き換えて得られる  $\lambda$  項を  $M[u \leftarrow N]$  で表す.

**補題 4.1**

$M, N \in \Lambda$ ,  $x \in \mathcal{V}$  のとき,  $M$  中の  $x$  のすべての出現位置  $u_1, u_2, \dots, u_n$  について,  $v \prec u_i$  かつ  $M/v \equiv (\lambda z. L) \implies z \notin \mathcal{V}(N)$  とすると,

$$M[x := N] =_{\alpha} M[u_1 \leftarrow N] \cdots [u_n \leftarrow N]$$

**補題 4.2 (代入補題)**

$L, M, N \in \Lambda$ ,  $x, y \in \mathcal{V}$ ,  $x \neq y$ ,  $x \notin \mathcal{V}(L)$  のとき,

$$M[x := N][y := L] =_{\alpha} M[y := L][x := N[y := L]]$$

**定義 (コンビネータ)**

自由変数を含まない  $\lambda$  項をコンビネータ, あるいは閉じた  $\lambda$  項という.

- (1)  $\mathbf{I} \equiv \lambda x. x$
- (2)  $\mathbf{K} \equiv \lambda x y. x \equiv \mathbf{T}$
- (3)  $\mathbf{F} \equiv \lambda x y. y$
- (4)  $\mathbf{S} \equiv \lambda x y z. x z (y z)$
- (5)  $\mathbf{B} \equiv \lambda x y z. x (y z)$
- (6)  $\mathbf{C} \equiv \lambda x y z. x z y$
- (7)  $\mathbf{\Omega} \equiv (\lambda x. x x)(\lambda x. x x)$

**4.4  $\beta$ 簡約****定義 (関係  $\beta$ )**

関係  $\beta$  を次のように定義する.

$$M \beta N \iff M \equiv (\lambda x. P) Q \text{ かつ } N \equiv P[x := Q]$$

**定義 ( $\beta$ 縮約,  $\beta$ 簡約)**

関係 $\rightarrow_\beta$ を次のように定義する.

- (1)  $M\beta N$  ならば  $M \rightarrow_\beta N$
- (2)  $M \rightarrow_\beta N$  ならば, 任意の  $L \in \Lambda$  について,  $(ML) \rightarrow_\beta (NL)$  かつ  $(LM) \rightarrow_\beta (LN)$
- (3)  $M \rightarrow_\beta N$  ならば, 任意の  $x \in \mathcal{V}$  について,  $(\lambda x.M) \rightarrow_\beta (\lambda x.N)$
- (4) (1)(2)(3) 以外に関係 $\rightarrow_\beta$ の成り立つものはない

両立性 (compatibility)

**定義 (推移的閉包)**

任意の集合  $X$  上の関係  $R$  について, 次のように  $R^n$  を定義する.

- $R^0 = \{\langle x, x \rangle \mid x \in X\}$
- $R^n = \{\langle x, z \rangle \mid \exists y \in X; xR^{n-1}y, yRz\} \quad (n > 0)$

また, 次のように定義される関係  $R^+$  を  $R$  の推移的閉包, 関係  $R^*$  を  $R$  の反射的推移的閉包という.

- $R^+ = \bigcup_{n>0} R^n$
- $R^* = \bigcup_{n \geq 0} R^n$

**定義 ( $\beta$ 簡約)**

関係 $\twoheadrightarrow_\beta$ を関係 $\rightarrow_\beta$ の反射的推移的閉包として定義する.

**定義 ( $\beta$ 簡約経路)**

$M_1 \rightarrow_\beta M_2, M_2 \rightarrow_\beta M_3, \dots, M_{n-1} \rightarrow_\beta M_n$  のとき, これを次のように表現する ( $\beta$ 簡約経路).

$$M_1 \rightarrow_\beta M_2 \rightarrow_\beta \dots \rightarrow_\beta M_n$$

**定義 ( $\beta$ 合同)**

関係 $=_\beta$ を次のように定義する.

- (1)  $M \twoheadrightarrow_\beta N \implies M =_\beta N$
- (2)  $M =_\beta N \implies N =_\beta M$
- (3)  $M =_\beta N$  かつ  $N =_\beta L \implies M =_\beta L$
- (4) (1)(2)(3) 以外に関係 $=_\beta$ の成り立つものはない

関係 $=_\beta$ は同値関係である.  $M =_\beta N$  であるとき,  $M$  と  $N$  は $\beta$ 合同であるという.

**4.5 正規形と Church-Rosser の定理****定義 ( $\beta$ 正規形)**

- (1)  $(\lambda x.P)Q$  なる形の項を $\beta$ 可簡約項 ( $\beta$ -redex) という
- (2) 部分項に $\beta$ 可簡約項を含まない $\lambda$ 項を $\beta$ 正規形という

$M \twoheadrightarrow_\beta N$  かつ  $N$  が $\beta$ 正規形であるとき,  $M$  は $\beta$ 正規形を持つという.

例

- $\Omega$ は $\beta$ 正規形を持たない
- $\mathbf{KI}\Omega$ は $\beta$ 正規形を持つ

$$\mathbf{KI}\Omega =_{\alpha} ((\lambda x.(\lambda y.x))\mathbf{I})\Omega \rightarrow_{\beta} (\lambda y.\mathbf{I})\Omega \rightarrow_{\beta} \mathbf{I}$$

しかし、次のようにして無限の $\beta$ 簡約も可能である。

$$\mathbf{KI}\Omega \rightarrow_{\beta} \mathbf{KI}\Omega \rightarrow_{\beta} \dots$$

**定理 (Church-Rosser の定理)**

任意の $\lambda$ 項  $M, N$  について,

$$M =_{\beta} N \implies \exists L (M \rightarrow_{\beta} L \text{ かつ } N \rightarrow_{\beta} L)$$

である。 $\rightarrow_{\beta}$ に限らず一般の簡約について、上を満たすものは Church-Rosser 性 (CR 性) を持つという。

**定理 (合流性)**

任意の $\lambda$ 項  $M, M_1, M_2$  について,

$$M \rightarrow_{\beta} M_1 \text{ かつ } M \rightarrow_{\beta} M_2 \implies \exists M_3 (M_1 \rightarrow_{\beta} M_3 \text{ かつ } M_2 \rightarrow_{\beta} M_3)$$

である。 $\rightarrow_{\beta}$ に限らず一般の簡約について、上を満たすものは合流性を持つという。

一般の簡約について、CR 性と合流性は同等であることが知られている。

系 4.1

任意の $\lambda$ 項は高々1つの $\beta$ 正規形しか持たない。

## 4.6 簡約の方法と戦略

(a) 簡約の方法

(b) 簡約の戦略

## 4.7 理論 $\Lambda$

## 4.8 帰納的関数論の理論 $\Lambda$ による解釈

$\lambda$ 定義可能と帰納的関数について説明する。

$\lambda$ 項での真理値の表現

$$\text{真} \equiv \mathbf{T} \equiv \lambda xy.x$$

$$\text{偽} \equiv \mathbf{F} \equiv \lambda xy.y$$

$\mathbf{T}MN =_{\beta} M, \mathbf{F}MN =_{\beta} N$  となる。

## λ項での対の表現

$$\langle M, N \rangle \equiv \lambda z. zMN$$

## λ項での自然数の表現

$$\begin{aligned} [0] &\equiv \mathbf{I} \equiv \lambda x. x \\ [n+1] &\equiv \langle \mathbf{F}, [n] \rangle \equiv \lambda z. z\mathbf{F}[n] \equiv \lambda z. z(\lambda xy. y)[n] \end{aligned}$$

## 定義 (λ定義可能)

$f: \mathcal{N}^n \rightarrow \mathcal{N}$  は、次を満たすコンビネータ  $F$  が存在するとき、λ定義可能であるという.

$$\forall x_1, \dots, x_n \in \mathcal{N}; F[x_1] \cdots [x_n] =_\beta [f(x_1, \dots, x_n)]$$

## 補題 4.4.1

初期関数  $z, s, u_i^{(n)}$  はλ定義可能である.

## 証明

$$\begin{aligned} C_0 &\equiv [0] \\ S^+ &\equiv \lambda x. \langle \mathbf{F}, x \rangle \\ U_i^n &\equiv \lambda x_1 \cdots x_n. x_i \end{aligned}$$

とおけばよい.

## 補題 4.4.2

λ定義可能な関数から、合成により定義した関数はλ定義可能である.

**証明**  $h, g_1, \dots, g_m$  がそれぞれλ項  $H, G_1, \dots, G_m$  によってλ定義可能とする. このとき,  $h, g_1, \dots, g_m$  から合成により次のように定義される関数  $f$

$$f(x_1, \dots, x_n) = h(g_1(x_1, \dots, x_n), \dots, g_m(x_1, \dots, x_n))$$

は、次のλ項によってλ定義可能である.

$$\lambda x_1 \cdots x_n. H(G_1 x_1 \cdots x_n) \cdots (G_m x_1 \cdots x_n)$$

**定理 4.9 (不動点定理)**

任意のλ項  $F$  について,  $FX =_{\beta} X$  となるλ項  $X$  が存在する. コンビネータ  $\mathbf{Y}$  を次のようにおくと, λ項  $\mathbf{Y}F$  はその一つである.

$$\mathbf{Y} \equiv \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$$

**証明**

$$\begin{aligned} \mathbf{Y}F &=_{\beta} \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))F \\ &=_{\beta} (\lambda x.F(xx))(\lambda x.F(xx)) \\ &=_{\beta} F((\lambda x.F(xx))(\lambda x.F(xx))) \end{aligned}$$

最後の2行を見ると

$$\mathbf{Y}F =_{\beta} F(\mathbf{Y}F)$$

**補題**

$$\mathbf{Z} \equiv \lambda x.x\mathbf{T}$$

とおいたとき,

$$\mathbf{Z}L M_1 M_2 =_{\beta} \begin{cases} M_1 & (L =_{\beta} [0] \text{ のとき}) \\ M_2 & (L =_{\beta} [n+1] \text{ のとき}) \end{cases}$$

**証明**

$$\begin{aligned} \mathbf{Z}[0] &=_{\beta} (\lambda x.x\mathbf{T})\mathbf{I} \\ &=_{\beta} \mathbf{IT} \\ &=_{\beta} \mathbf{T} \\ \mathbf{Z}[n+1] &=_{\beta} (\lambda x.x\mathbf{T})(\lambda z.z\mathbf{F}[n]) \\ &=_{\beta} (\lambda z.z\mathbf{F}[n])\mathbf{T} \\ &=_{\beta} \mathbf{TF}[n] \\ &=_{\beta} \mathbf{F} \end{aligned}$$

であり,  $\mathbf{T}M_1M_2 =_{\beta} M_1$ ,  $\mathbf{F}M_1M_2 =_{\beta} M_2$  である.

**補題**

$$\mathbf{P}^- \equiv \lambda x.x\mathbf{F}$$

とおいたとき,

$$\mathbf{P}^-L =_{\beta} \begin{cases} \mathbf{F} & (L =_{\beta} [0] \text{ のとき}) \\ [n] & (L =_{\beta} [n+1] \text{ のとき}) \end{cases}$$

証明

$$\begin{aligned}
P^- [0] &=_{\beta} (\lambda x.x\mathbf{F})\mathbf{I} \\
&=_{\beta} \mathbf{IF} \\
&=_{\beta} \mathbf{F} \\
P^- [n+1] &=_{\beta} (\lambda x.x\mathbf{F})(\lambda z.z\mathbf{F} [n]) \\
&=_{\beta} (\lambda z.z\mathbf{F} [n])\mathbf{F} \\
&=_{\beta} \mathbf{FF} [n] \\
&=_{\beta} [n]
\end{aligned}$$

である.

### 補題 4.4.3

$\lambda$ 定義可能な関数から, 原始帰納法により定義した関数は $\lambda$ 定義可能である.

**証明**  $h, g$  がそれぞれ $\lambda$ 項  $H, G$  によって $\lambda$ 定義可能とする.  $h, g$  から原始帰納法により次のように定義される関数  $f$  (ただし,  $\vec{y}$  は  $y_1, \dots, y_n$  の略記)

$$\begin{aligned}
f(0, \vec{y}) &= g(\vec{y}) \\
f(x', \vec{y}) &= h(x, f(x, \vec{y}), \vec{y})
\end{aligned}$$

は, 次の $\lambda$ 項  $F$  によって $\lambda$ 定義可能である.

$$\begin{aligned}
F &\equiv \mathbf{YN} \\
N &\equiv \lambda f x \vec{y}. Z x M_1 M_2 \\
M_1 &\equiv G \vec{y} \\
M_2 &\equiv H(P^- x)(f(P^- x)\vec{y})\vec{y}
\end{aligned}$$

なぜなら

$$Z x M_1 M_2 =_{\beta} \begin{cases} G \vec{y} & (x =_{\beta} [n] \text{ のとき}) \\ H [n] (f [n] \vec{y}) \vec{y} & (x =_{\beta} [n+1] \text{ のとき}) \end{cases}$$

例

plus は次の $\lambda$ 項  $F$  によって $\lambda$ 定義可能である.

$$\begin{aligned}
F &\equiv \mathbf{YN} \\
N &\equiv \lambda f x y. Z x y (S^+(f(P^- x)y))
\end{aligned}$$

### 補題 4.4.4

$\lambda$ 定義可能な関数から, 最小化により定義した関数は $\lambda$ 定義可能である.

**証明**  $g$  が  $\lambda$  項  $G$  によって  $\lambda$  定義可能とする.  $g$  から最小化により次のように定義される関数

$$f(\vec{x}) = \mu y [g(y, \vec{x})]$$

は, 次の  $\lambda$  項  $F$  によって  $\lambda$  定義可能である.

$$\begin{aligned} F &\equiv H [0] \\ H &\equiv \mathbf{Y}H_1 \\ H_1 &\equiv \lambda h y \vec{x}. Z(Gy\vec{x})(h(S^+y)\vec{x})y \end{aligned}$$

#### 補題 4.5

帰納的部分関数は  $\lambda$  定義可能である.

#### 定理 4.10 (Kleene)

$\lambda$  定義可能な数値関数のクラスと帰納的部分関数のクラスは一致する.

**証明** 省略

#### 4.9 他の簡約の概念

#### 4.10 コンビネータ論理

#### 4.11 関数型プログラムとラムダ計算の関係

#### 4.12 ラムダ計算の集合論的モデル

## 5 論理モデル

### 5.1 基本的な考え方

### 5.2 一階述語論理の言語

### 5.3 一階述語論理の意味

### 5.4 証明の手続き

### 5.5 論理モデルにおける計算可能性

## 6 書換えモデル

### 6.1 基本的な考え方

### 6.2 抽象書換え系

#### 定義 (抽象書換え系)

任意の集合  $A$  と  $A$  上の任意の関係  $\rightarrow$  の対  $(A, \rightarrow)$  を抽象書換え系 (あるいは簡約系, reduction system) という.

#### 定義 ( $\rightarrow^+$ , $\rightarrow\rightarrow$ , $\leftarrow$ , $\leftarrow^+$ , $\leftarrow\leftarrow$ , $\downarrow$ )

- $\rightarrow$  の推移的閉包を  $\rightarrow^+$  で表す.
- $\rightarrow$  の反射的推移的閉包を  $\rightarrow\rightarrow$  で表す.
- $s \rightarrow t$ ,  $s \rightarrow^+ t$ ,  $s \rightarrow\rightarrow t$  を, それぞれ  $t \leftarrow s$ ,  $t \leftarrow^+ s$ ,  $t \leftarrow\leftarrow s$  とも書く.
- $b \downarrow c$  により  $(\exists a \in A; b \rightarrow\rightarrow a \leftarrow\leftarrow c)$  を表す.

#### 定義 (簡約経路)

$a_0 \rightarrow a_1 \rightarrow \cdots \rightarrow a_n$  ( $n \geq 0$ ) を  $a_0$  から  $a_n$  への簡約経路,  $n$  を簡約経路の長さという.

#### 定義 (弱合流性)

$\rightarrow$  が, 任意の  $a, b, c \in A$  に対して次の条件を満たすとき,  $\rightarrow$  は弱合流性 (WCR 性) を持つという.

$$b \leftarrow a \rightarrow c \implies b \downarrow c$$

#### 定義 (合流性)

$\rightarrow$  が, 任意の  $a, b, c \in A$  に対して次の条件を満たすとき,  $\rightarrow$  は合流性を持つ, あるいは  $\rightarrow\rightarrow$  はダイヤモンド性を持つという.

$$b \leftarrow\leftarrow a \rightarrow\rightarrow c \implies b \downarrow c$$

#### 補題 6.1

$$\rightarrow \text{が合流性を持つ} \iff \forall a, b, c \in A (b \leftarrow a \rightarrow\rightarrow c \implies b \downarrow c)$$

#### 証明

( $\implies$ ) 合流性の定義より明らか.

( $\impliedby$ ) 教科書の図 6.4 を用いた図式追跡法による.

**定義 (=)**

$A$  上の関係  $=$  を次のように定義する. 任意の  $a, b, c \in A$  に対して,

- (1)  $a \rightarrow b \implies a = b$
- (2)  $a = a$
- (3)  $a = b \implies b = a$
- (4)  $a = b$  かつ  $b = c \implies a = c$
- (5) (1)(2)(3)(4) 以外に関係  $=$  の成り立つものはない

関係  $=$  は同値関係である. なお,  $a$  と  $b$  が同一の要素であることは  $a \equiv b$  で表す.

**定義 (CR 性)**

$\rightarrow$  が, 任意の  $a, b \in A$  に対して次の条件を満たすとき,  $\rightarrow$  は Church-Rosser 性 (CR 性) を持つという.

$$a = b \implies a \downarrow b$$

**補題 6.2**

$$\rightarrow \text{が合流性を持つ} \iff \rightarrow \text{が CR 性を持つ}$$

**定義 (正規形)**

- $a \in A$  が正規形である:  $a \rightarrow b$  なる  $b \in A$  が存在しないとき
- $a \in A$  が正規形を持つ:  $a \rightarrow b$  なる正規形  $b \in A$  が存在するとき

**定義 (抽象書換え系の性質)**

抽象書換え系  $A = (A, \rightarrow)$  について,

- (2) 弱正規化性 (Weakly Normalizing, WN 性): すべての  $a \in A$  が正規形を持つとき
- (3) 強正規化性 (Strongly Normalizing, SN 性, 停止性, 有限停止性, Noetherian, ネーター的): すべての簡約経路が有限であるとき
- (4) UN 性 (Uniform Normal form property): 任意の  $a, b, c \in A$  について対が成り立つとき

$$b \leftarrow a \rightarrow c \text{ かつ } b, c \text{ が正規形} \implies b \equiv c$$

- (5) NF 性 (Normal Form property): 任意の  $a, b \in A$  について対が成り立つとき

$$a \text{ が正規形かつ } a = b \implies b \rightarrow a$$

**定理 6.1**

- (1)  $CR \implies NF$
- (2)  $NF \implies UN$
- (3)  $CR \implies UN$
- (4)  $UN \wedge WN \implies CR$
- (5)  $SN \wedge WCR \implies CR$  (Newman の補題)

**定義 (完備)**

抽象書換え系  $A$  が CR 性と SN 性を持つとき、 $A$  は完備 (complete) であるという。

**例 (半 Thue 系, 半トウエ系, STS)**

半 Thue 系  $(X, S)$  は、有限個の記号からなる集合  $X$  (アルファベットと呼ばれる) と、 $X^*$  上の二項関係  $S$  (記号列の書換え規則を表す) の対である。ここで、 $X^*$  は  $X$  の要素の列全体 (空語を含む) を表す。

半 Thue 系  $(X, S)$  は、 $\rightarrow$  を次のように定義することにより、抽象書換え系  $(X^*, \rightarrow)$  とみなせる。

$$\rightarrow = \{(xuy, xvy) \mid (u, v) \in S, x, y \in X^*\}$$

**6.3 項書換え系**

項書換え系 (Term Rewriting System, TRS) の定義の準備を行なう。

**定義 (変数, 関数記号)**

- $\mathcal{V}$ : 加算無限個の変数  $(x, y, z, \dots)$  の集合
- $\mathcal{F}$ : 有限個の関数記号の集合。各関数記号はあらかじめ定められた項数 (arity) を持つ。項数が 0 の関数記号を定数と呼ぶ。

**定義 (項)**

$\mathcal{V}$  を変数の集合、 $\mathcal{F}$  を関数記号の集合としたとき、以下のように定義される項 (term) の集合を  $\mathcal{T}(\mathcal{F}, \mathcal{V})$  で表す。

- (1) 変数  $x \in \mathcal{V}$  は項である。
- (2)  $f \in \mathcal{F}$  が項数  $n$  の関数記号で、 $t_1, t_2, \dots, t_n$  が項のとき、 $f(t_1, t_2, \dots, t_n)$  は項である。特に  $n = 0$  のとき、 $f()$  を単に  $f$  と書く。
- (3) (1)(2) で定義されるものだけが項である。

**定義**

- 項  $t$  に含まれる変数の集合を  $\mathcal{V}(t)$  で表す。
- 変数を含まない項 (すなわち  $\mathcal{V}(t) = \emptyset$  なる項  $t$ ) を閉じた項 (closed term) または基礎項 (ground term) と呼ぶ。
- 閉じた項の集合を  $\mathcal{T}(\mathcal{F})$  で表す。

**定義 (代入)**

$\mathcal{V}$  から  $\mathcal{T}(\mathcal{F}, \mathcal{V})$  への関数を代入という。代入  $\theta$  のは、次のように  $\mathcal{T}(\mathcal{F}, \mathcal{V})$  から  $\mathcal{T}(\mathcal{F}, \mathcal{V})$  への関数  $\bar{\theta}$  に拡張される。

- (1) 変数  $x$  について、 $\bar{\theta}(x) = \theta(x)$
- (2) 項  $f(t_1, t_2, \dots, t_n)$  が項のとき、 $\bar{\theta}(f(t_1, t_2, \dots, t_n)) = \theta(f(\bar{\theta}(t_1), \bar{\theta}(t_2), \dots, \bar{\theta}(t_n)))$

以降は、 $\bar{\theta}$ も単に $\theta$ と書く。また、代入を項と変数を“/”で区切った順序対の集合

$$\{t_1/x_1, t_2/x_2, \dots, t_n/x_n\}$$

で表すことがある。また、代入 $\theta$ と $\psi$ の合成を $\theta\psi$ で表す(すなわち、 $\theta\psi(t) = \theta(\psi(t))$ )。

### 定義 (単一化)

- 項  $s, t$  に対して  $\theta(s) = \theta(t)$  のとき、代入  $\theta$  は  $s$  と  $t$  の単一化代入 (unifier) と呼ばれる。また  $s$  と  $t$  は単一化可能であるといわれる。
- $\theta$  を項  $s$  と  $t$  の単一化代入とする。このとき、任意の単一化代入  $\nu$  に対して  $\nu = \gamma\theta$  となる代入  $\gamma$  が存在するならば、 $\theta$  は  $s$  と  $t$  の最も一般的な代入 (most general unifier, mgu) と呼ばれる。

### 定義 (文脈)

□ を欠落 (slot, hole) と呼び、文脈 (context) を以下のように定義する。

- (1) 変数と欠落は文脈である。
- (2)  $f$  が項数  $n$  の関数記号で、 $t_1, t_2, \dots, t_n$  が文脈のとき、 $f(t_1, t_2, \dots, t_n)$  は文脈である。特に  $n = 0$  のとき、 $f()$  を単に  $f$  と書く。
- (3) (1)(2) で定義されるものだけが文脈である。

ただし、以下では欠落を 1 個だけ持つ文脈だけを考える。また、 $C[]$  など文脈を表し、 $C[t]$  により  $C[]$  の欠落を項  $t$  で置き換えた項を表す。

### 定義 (項書換え系)

項書換え系 (Term Rewriting System, TRS) は、項の集合  $\mathcal{T}$  と  $\mathcal{T}$  上の書換え規則の有限集合  $R$  の対  $(\mathcal{T}, R)$  で規定される。 $\mathcal{T}$  上の書換え規則は二つの項を記号  $\Rightarrow$  で結んだものである。

項書換え系  $(\mathcal{T}, R)$  は、 $\rightarrow$  を次のように定義した抽象書換え系  $(\mathcal{T}, \rightarrow)$  を意味する。

$$\begin{aligned} \rightarrow = \{ & (a, b) \mid \text{文脈 } C[], \text{ 代入 } \theta, \text{ 規則 } s \Rightarrow t \in R \text{ が存在し,} \\ & a = C[\theta(s)], b = C[\theta(t)] \} \end{aligned}$$

例 6.3  $\mathcal{F} = \{p, m, s, 0\}$  (項数はそれぞれ 2, 2, 1, 0),  $\mathcal{V} = \{x, y, \dots\}$ , また

$$R = \{ \begin{array}{l} p(x, 0) \Rightarrow x, \\ p(x, s(y)) \Rightarrow s(p(x, y)) \\ m(x, 0) \Rightarrow 0 \\ m(x, s(y)) \Rightarrow p(m(x, y), x) \end{array} \}$$

としたとき,  $(\mathcal{T}(\mathcal{F}, \mathcal{V}), R)$  は項書換え系の例である.

$$\begin{array}{l} p(m(s(0), s(0)), 0) \\ \rightarrow p(p(m(s(0), 0), s(0)), 0) \\ \rightarrow p(p(0, s(0)), 0) \\ \rightarrow p(s(p(0, 0)), 0) \\ \rightarrow p(s(0), 0) \\ \rightarrow s(0) \end{array}$$

## 6.4 正則項書換え系

### 定義 (線形)

- 項  $t$  が線形: 同一変数を 2 つ以上含まないとき
- 書換え規則  $s \Rightarrow t$  が左線形:  $s$  が線形するとき
- 項書換え系が左線形: すべての書換え規則が左線形

### 定義 (項の重なり)

2 つの書換え規則  $s_1 \Rightarrow t_1, s_2 \Rightarrow t_2$  について次の条件を満たすとき,  $s_2 \Rightarrow t_2$  は  $s_1 \Rightarrow t_1$  に重なるという.

$$\begin{array}{l} \exists u \in \mathcal{O}(s_1) \exists \theta; s_1/u \notin \mathcal{V} \text{ かつ } \theta(s_1/u) \equiv \theta(s_2) \\ \text{(ただし } s_1 \Rightarrow t_1 \text{ と } s_2 \Rightarrow t_2 \text{ が同一のときは } u \neq \epsilon \text{)} \end{array}$$

### 定義 (正則項書換え系)

左線形かつ重なりのない項書換え系を正則項書換え系という.

### 定理 6.2

正則項書換え系は CR 性を持つ.

## 6.5 項書換えの簡約戦略

## 6.6 危険対

### 定義 (危険対)

$r_1$  と  $r_2$  をそれぞれ  $s_1 \Rightarrow t_1$ ,  $s_2 \Rightarrow t_2$  なる書換え規則とし,  $r_2$  が出現位置  $u \in \mathcal{O}(s_1)$  で mgu  $\theta$  により  $r_1$  に重なるとする. このとき次のような項の対を  $r_1$  と  $r_2$  の危険対という.

$$\langle \theta(s_1[u \leftarrow t_2]), \theta(t_1) \rangle$$

項書換え系  $(\mathcal{T}, R)$  が重なりを持たないということは, すべての書換え規則  $r_1, r_2 \in R$  に対し  $r_1$  と  $r_2$  の危険対が存在しないことと同値である.

### 定義 (危険対の集合)

任意の書換え規則  $r_1, r_2 \in R$  について, 集合  $CP_{\langle r_1, r_2 \rangle}$  を次のように定義する.

$$CP_{\langle r_1, r_2 \rangle} = \{ \langle p, q \rangle \mid r_1 \text{ と } r_2 \text{ の危険対 } \langle p, q \rangle \}$$

また  $r_1$  と  $r_2$  の危険対の集合を  $CP_{\langle r_1, r_2 \rangle} \cup CP_{\langle r_2, r_1 \rangle}$  で定義し,  $R$  の危険対の集合  $CP$  を  $CP = \bigcup_{r_1, r_2 \in R} CP_{\langle r_1, r_2 \rangle}$  で定義する.

### 定理 6.4

$\mathcal{R} = (\mathcal{T}, R)$  が項書換え系,  $CP$  が  $R$  の危険対の集合のとき

$$\mathcal{R} \text{ が WCR 性を持つ} \iff \forall \langle p, q \rangle \in CP; p \downarrow q$$

### 定理 6.5

$\mathcal{R} = (\mathcal{T}, R)$  が SN 性を持つ項書換え系,  $CP$  が  $R$  の危険対の集合のとき

$$\mathcal{R} \text{ が CR 性を持つ} \iff \forall \langle p, q \rangle \in CP; \hat{p} \equiv \hat{q}$$

ここで  $\hat{t}$  は  $t$  の正規系である.

SN である項書換え系  $\mathcal{R} = (\mathcal{T}, R)$  の CR 性は以下の方法で証明できる.

- $R$  のすべての危険対  $\langle p, q \rangle$  について,  $\hat{p} \equiv \hat{q}$  であれば  $\mathcal{R}$  は CR である.

## 6.7 完備化手続き

(a) 背景

(b) 等式理論

普遍代数における語の問題 (word problem)

$$\left. \begin{array}{l} 0 + x = x \\ (-x) + x = 0 \\ (x + y) + z = x + (y + z) \end{array} \right\} \implies x + (-x) = 0 \quad ?$$

(c) 完備化手続き

定義 (厳格半順序)

$\mathcal{T}$  上の関係  $\succ$  が, 推移的かつ非反射的であり, また  $t_1 \succ t_2 \succ \dots$  となるような  $\mathcal{T}$  の無限個の要素の列  $t_1, t_2, \dots$  が存在しないとき,  $\succ$  は  $\mathcal{T}$  上の厳格半順序とよばれる.

CPC (Critical Pair Completion) 手続き

項書換え系  $\mathcal{R} = (\mathcal{T}, R)$  について,  $\mathcal{T}$  上の厳格半順序  $\succ$  が存在し,  $R$  のすべての規則  $s \Rightarrow t$  について  $s \succ t$  とする. このとき,  $\mathcal{R}$  に対する CPC 手続きは以下のようなになる.

- (1)  $CP \leftarrow R$  の危険対の集合
- (2)  $CP = \emptyset$  であれば完備化成功
- (3) 任意の対  $\langle p, q \rangle$  を  $CP$  から取り出す
- (4)  $p, q$  の  $R$  に関する正規系  $\hat{p}, \hat{q}$  を求める
- (5)  $\hat{p} \equiv \hat{q}$  のとき (2) へ
- (6)  $\hat{p} \succ \hat{q}$  のとき,  $\langle \alpha, \beta \rangle \leftarrow \langle \hat{p}, \hat{q} \rangle$  とする
- (7)  $\hat{q} \succ \hat{p}$  のとき,  $\langle \alpha, \beta \rangle \leftarrow \langle \hat{q}, \hat{p} \rangle$  とする
- (8) どれでもなければ完備化失敗
- (9)  $R$  に  $\alpha \Rightarrow \beta$  を付け加える
- (10) 新たに生じる危険対の集合を  $CP$  に付け加える
- (11) (2) へ

例 次の項書換え系  $\mathcal{R} = (\mathcal{T}(\mathcal{F}, \mathcal{V}), R)$  を完備化する.  $\succ$  は項の長さの比較 (より長い) とする.

$$\begin{aligned} \mathcal{F} &= \{ \cdot, W, B, S, \text{nil} \} \\ \mathcal{V} &= \{ x, y, \dots \} \\ R &= \left\{ \begin{array}{l} W \cdot (B \cdot x) \Rightarrow B \cdot x, \\ B \cdot (S \cdot x) \Rightarrow W \cdot x \end{array} \right\} \end{aligned}$$

- (1)  $CP \leftarrow \{ \langle W \cdot (W \cdot x), B \cdot (S \cdot x) \rangle \}$   
 (3)  $\langle p, q \rangle \leftarrow \langle W \cdot (W \cdot x), B \cdot (S \cdot x) \rangle, CP \leftarrow \emptyset$   
 (4)  $\hat{p} = W \cdot (W \cdot x), \hat{q} = W \cdot x$   
 (6)  $\langle \alpha, \beta \rangle \leftarrow \langle W \cdot (W \cdot x), W \cdot x \rangle$   
 (9)  $R \leftarrow R \cup \{ W \cdot (W \cdot x) \Rightarrow W \cdot x \}$   
 (10)  $CP \leftarrow \{ \langle W \cdot (B \cdot x), W \cdot (B \cdot x) \rangle, \langle W \cdot (W \cdot x), W \cdot (W \cdot x) \rangle \}$   
 (3)  $\langle p, q \rangle \leftarrow \langle W \cdot (B \cdot x), W \cdot (B \cdot x) \rangle, CP \leftarrow \{ \langle W \cdot (W \cdot x), W \cdot (W \cdot x) \rangle \}$   
 (3)  $\langle p, q \rangle \leftarrow \langle W \cdot (W \cdot x), W \cdot (W \cdot x) \rangle, CP \leftarrow \emptyset$   
 (2) 完備化成功

(d) メタ推論系としての完備化手続き

## 6.8 項書換え系の停止性

## 7 代数モデル

### 7.1 基本的な考え方

### 7.2 指標と $\Sigma$ 代数

### 7.3 仕様と $\Omega$ 代数