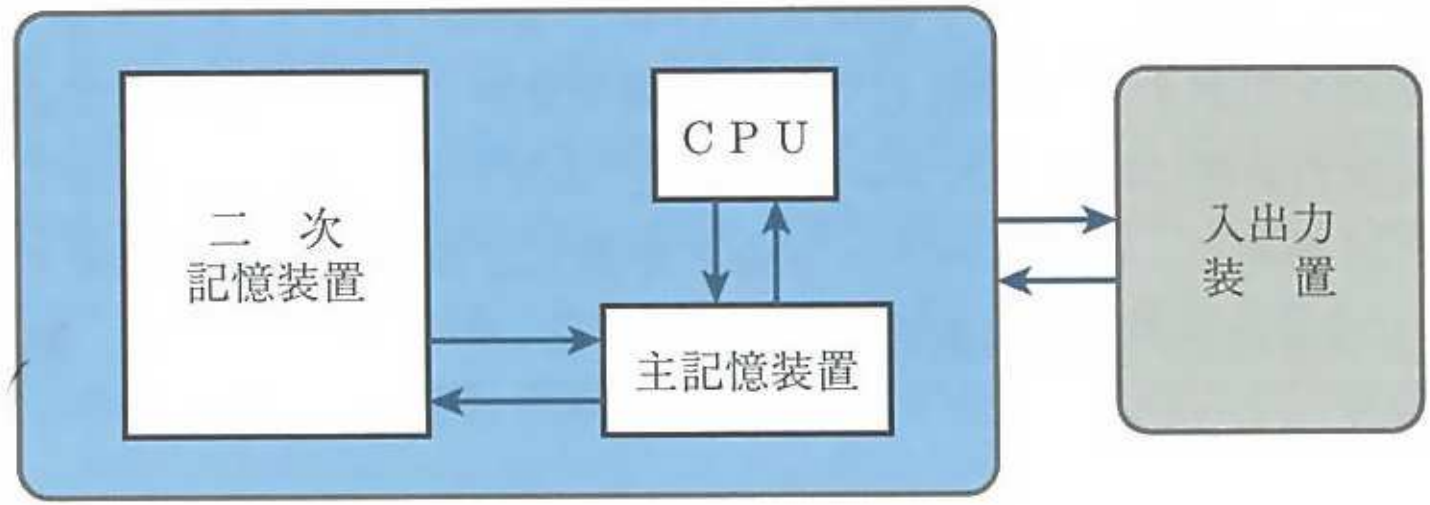
5 計算のモデル

5.1,5.2 コンピュータ・CPUの動作原理

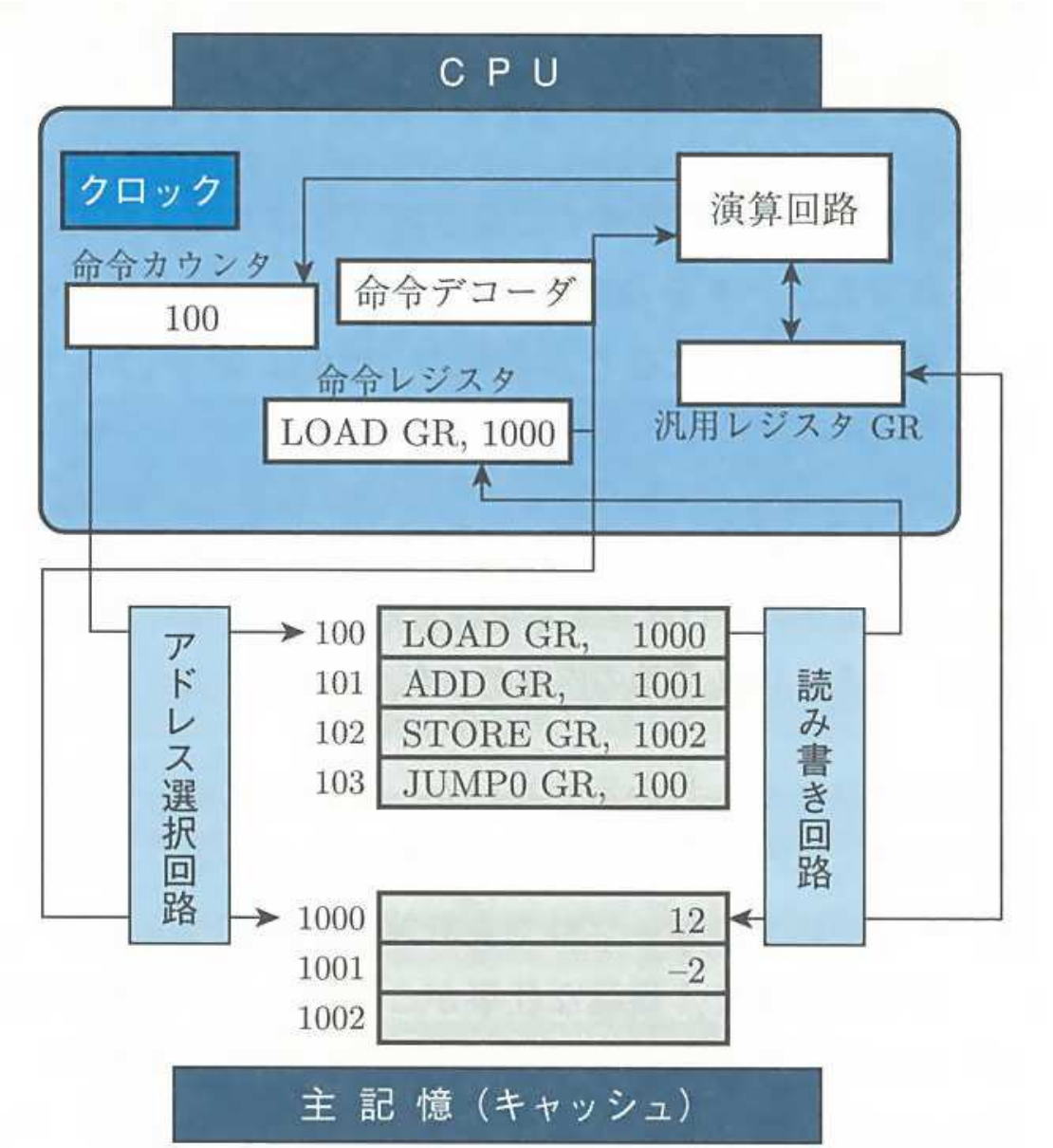
5.3 計算の数学的モデル

5.1,5.2 コンピュータ・CPUの動作原理



ストアードメモリ方式： プログラムもデータとして格納

- 機械語
- レジスタ


CPU
クロック
演算回路
命令カウンタ
100
命令デコーダ
命令レジスタ
LOAD GR, 1000
汎用レジスタ GR
アドレス選択回路
100 LOAD GR, 1000
101 ADD GR, 1001
102 STORE GR, 1002
103 JUMP0 GR, 100
読み書き回路
1000 12
1001 −2
1002
主記憶（キャッシュ）

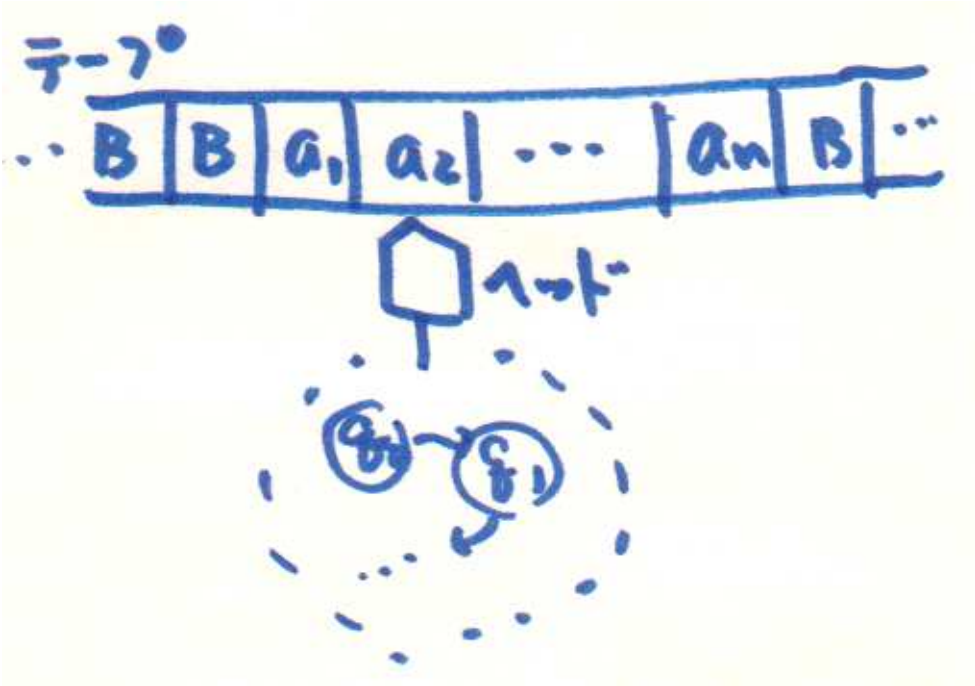
5.3 計算の数学的モデル

- チューリング機械
- レジスタ機械、
- λ計算、
- 項書換え系、

などなど

5.3.1 チューリング機械

- 無限テープ
- テープの記号と状態から動作を(非決定的に)決める
- 動作は、次の状態・テープに書き込む記号・ヘッドの移動

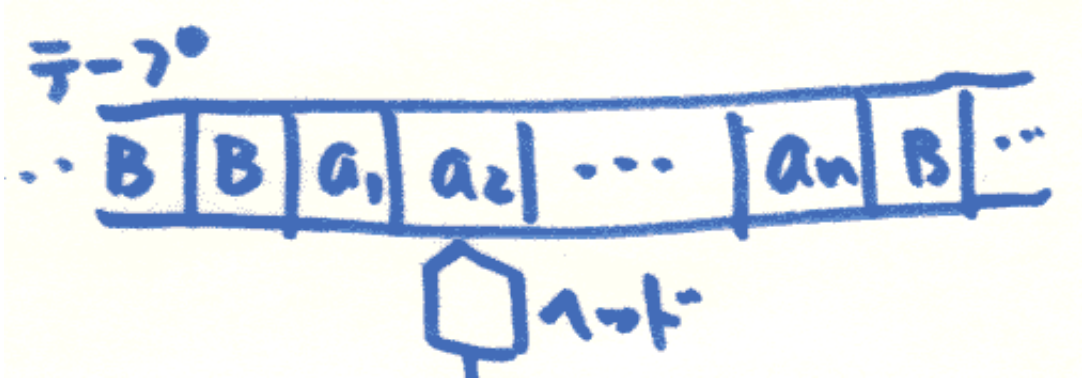


チューリング機械(TM)：

$$M = (Q, \Sigma, \Gamma, \delta, q_0)$$

- Q: 状態の有限集合
- Σ: 入出力記号の有限集合
- Γ: テープ記号の有限集合　$(\Gamma \supseteq \Sigma \cup \{B\})$
- δ: 遷移関数$(\delta : \ Q \times \Gamma \to Q \times (\Gamma \cup \{L, R\}))$
- q_0: 初期状態 $(q_0 \in Q)$

時点表示： 状態がqでヘッドの位置が以下のとき、



$$a_1 q a_2 \cdots a_n$$

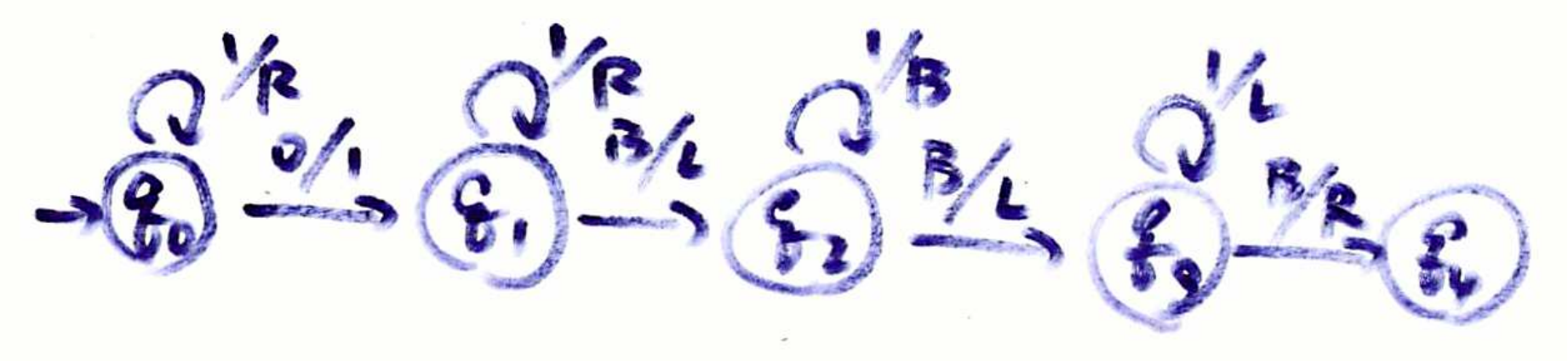
と書く

チューリング機械が定義する自然数上の計算：

$$q_0 1^{k_1} 0 1^{k_2} 0 \cdots 0 1^{k_n} \vdash^* q 1^m$$

は$f(k_1, \ldots, k_n) = m$

例： $f(x,y) = x + y$を満たす$f : \mathbb{N}^2 \to \mathbb{N}$を計算する
チューリング機械



- 入力1011に対する動作
  $q_0 1011 \vdash 1 q_0 011 \vdash 1 q_1 111 \vdash 11 q_1 11 \vdash$
  $111 q_1 1 \vdash 1111 q_1 \vdash 111 q_2 1 \vdash 111 q_2 \vdash$
  $11 q_3 1 \vdash 1 q_3 11 \vdash q_3 111 \vdash q_3 B111 \vdash$
  $q_4 111$

5.3.2 帰納的関数

原始帰納的関数： 自然数$\mathbb{N}$上の関数のクラス

原始帰納的関数の定義

- 零関数、次者関数、射影関数

  zero: $\mathbb{N}^0 \to \mathbb{N}$ ただし、**zero**() $\stackrel{\text{def}}{=} 0$

  suc: $\mathbb{N} \to \mathbb{N}$ ただし、**suc**$(x) \stackrel{\text{def}}{=} x + 1$

  $\mathbf{p}_i^n : \mathbb{N}^n \to \mathbb{N}$ ただし、$\mathbf{p}_i^n(x_1, \ldots, x_n) \stackrel{\text{def}}{=} x_i$

- 合成関数、すなわち、$g : \mathbb{N}^m \to \mathbb{N}$と$g_j : \mathbb{N}^n \to \mathbb{N}$が原始帰納的関数のとき、

  $$f(\vec{x}) = g(g_1(\vec{x}), \ldots, g_m(\vec{x}))$$

  で定義される関数$f : \mathbb{N}^n \to \mathbb{N}$は原始帰納的関数

原始帰納的関数の定義(続き)

- 原始帰納法で定義された関数、すなわち、

  $g : \mathbb{N}^n \to \mathbb{N}$ と $h : n^{n+2} \to \mathbb{N}$ が原始帰納的関数のとき、

  $$f(\vec{x}, 0) = g(\vec{x})$$

  $$f(\vec{x}, y+1) = h(\vec{x}, y, f(\vec{x}, y))$$

  で定義される関数 $f : \mathbb{N}^{n+1} \to \mathbb{N}$ は原始帰納的関数

原始帰納的関数の例

- $\mathbf{one}() \stackrel{\mathbf{def}}{=} 1$ なぜなら $1 = \mathbf{suc}(\mathbf{zero}())$
- $\mathbf{two}() \stackrel{\mathbf{def}}{=} 2$ なぜなら $2 = \mathbf{suc}(\mathbf{one}())$
- $\mathbf{pred}(x) \stackrel{\mathbf{def}}{=} x \dot{-} 1$ なぜなら

  $\mathbf{pred}(0) = 0$ $\mathbf{(= zero())}$
  $\mathbf{pred}(y+1) = y$ $\mathbf{(= p_1^2(}y, \mathbf{pred}(y)\mathbf{))}$
- $\mathbf{add}(x,y) \stackrel{\mathbf{def}}{=} x + y$ なぜなら

  $\mathbf{add}(x,0) = x$ $\mathbf{(= p_1^1(}x\mathbf{))}$
  $\mathbf{add}(x,y+1) = \mathbf{suc}(\mathbf{add}(x,y))$
  $\mathbf{(= h(}x,y,\mathbf{add}(x,y)\mathbf{))}$

  ここで、$\mathbf{h}(x,y,z) = \mathbf{suc}(\mathbf{p}_3^3(x,y,z))$ は原始帰納的

原始帰納的関数の例(続き)

- $x \dot{-} y = \begin{cases} x - y & \textbf{if } x >= y \\ 0 & \textbf{otherwise} \end{cases}$
  を計算する関数$\textbf{sub}(x, y)$ なぜなら

  $\textbf{sub}(x, 0) = x$
  $\textbf{sub}(x, y + 1) = \textbf{prod}(\textbf{sub}(x, y))$

- $x \times y$を計算する関数$\textbf{mult}(x, y)$ なぜなら

  $\textbf{mult}(x, 0) = 0$
  $\textbf{mult}(x, y + 1) = \textbf{add}(x, \textbf{mult}(x, y))$

問： 関数$\textbf{exp}(x, y) = x^y$、$\textbf{fact}(x) = x!$、$\textbf{min}(x, y)$が原始帰納的であることを示せ。(ここで0^0の値は問わない)

補題： 原始帰納的関数$f(\vec{x}, y)$を利用して次のように定義される$f'(\vec{x}, y)$と$f''(\vec{x}, y)$は原始帰納的関数である

$$f'(\vec{x}, y) = \sum_{z<y} f(\vec{x}, z) = f(\vec{x}, 0) + \cdots + f(\vec{x}, y-1)$$

$$f''(\vec{x}, y) = \prod_{z<y} f(\vec{x}, z) = f(\vec{x}, 0) \times \cdots \times f(\vec{x}, y-1)$$

証明： f'のみ示し、f''は省略

$$f'(\vec{x}, 0) = 0$$
$$f'(\vec{x}, y+1) = f'(\vec{x}, y) + f(\vec{x}, y)$$

定理 原始帰納的関数はチューリング機械で計算できる

略証： 原始帰納的関数の構成に関する帰納法

- **zero**()，**suc**()，$\mathbf{p}_i^n(x_1, \ldots, x_n)$はチューリング機械で計算できる
- 合成

  $g : \mathbb{N}^m \to \mathbb{N}$と$g_j : \mathbb{N}^n \to \mathbb{N}$を計算するチューリング機械があると仮定し、$g(g_1(\vec{x}), \ldots, g_m(\vec{x}))$はチューリング機械で計算できることを示す

略証 (続き)

- 原始帰納法

  $g : \mathbb{N}^n \to \mathbb{N}$ と $h : \mathbb{N}^{n+2} \to \mathbb{N}$ を計算するチューリング機械があると仮定し、以下で定義される f がチューリング機械で計算できることを示す

$$\begin{cases} f(\vec{x}, 0) = g(\vec{x}) \\ f(\vec{x}, y+1) = h(\vec{x}, y, f(\vec{x}, y)) \end{cases}$$

原始帰納的述語

- **(n変数)** 述語**:** $p : \mathbb{N}^n \to \{\text{真}, \text{偽}\}$
- 述語pの特徴関数$c_p : \mathbb{N}^n \to \mathbb{N}$

$$c_p(\vec{x}) = \begin{cases} 0 & \cdots \quad p(\vec{x}) = \text{真のとき} \\ 1 & \cdots \quad p(\vec{x}) = \text{偽のとき} \end{cases}$$

- 述語pが原始帰納的 $\overset{\textbf{def}}{\Longleftrightarrow}$ c_pが原始帰納的

例： 述語「$=0$」 **($=0(x)$を$x = 0$と書くことにする)**

$$c_{=0}(x) = \begin{cases} 0 & \cdots \quad x\text{が}0\text{のとき} \\ 1 & \cdots \quad x\text{が}0\text{でないとき} \end{cases}$$

$c_{=0}(x) = 1 \dot{-} (1 \dot{-} x)$と書けるので、$c_{=0}$は原始帰納的関数である。よって「$=0$」は原始帰納的述語

例： 述語「$x = y$」、「$x \leq y$」は原始帰納的。なぜなら、

$$c_{=}(x, y) = c_{=0}((x \dot{-} y) + (y \dot{-} x))$$

$$c_{\geq}(x, y) = c_{=0}(x \dot{-} y)$$

補題 $p(\vec{x})$, $q(\vec{x})$, $r(\vec{x}, y)$ が原始帰納的述語なら、次の述語も原始帰納的

(1) $\neg p(\vec{x})$　　**(2)** $p(\vec{x}) \vee q(\vec{x})$　　**(3)** $p(\vec{x}) \wedge q(\vec{x})$

(4) $(\exists z < y)\, r(\vec{x}, z)$　　**(5)** $(\forall z < y)\, r(\vec{x}, z)$

(6) $p(f_1(\vec{x}), \ldots, f_n(\vec{x}))$ **(ここで** f_i **は原始帰納的)**

証明：**(3),(5)** はそれぞれ **(2),(4)** で表せる

(1) $c_{\neg p}(\vec{x}) = 1 \dot{-} c_p(\vec{x})$

(2) $c_{p \vee q}(\vec{x}) = c_p(\vec{x}) \times c_q(\vec{x})$

(4) 述語を $s(\vec{x}, y)$ とかく。$c_s(\vec{x}, y) = \prod_{z<y} c_r(\vec{x}, z)$

(6) 述語を $p'(\vec{x})$ とかく。

$$c_{p'}(\vec{x}) = c_p(f_1(\vec{x}), \ldots, f_n(\vec{x}))$$

例： 次の述語は原始帰納的

- $x < y$
- $x \times y = z$
- **prime**(x) $\overset{\textbf{def}}{\Longleftrightarrow}$

  $(x > 1) \wedge (\neg(\exists u < x)(\exists v < x)\ (u \times v = x))$

$p(\vec{x}, y)$の限定最小解関数： y未満で$p(\vec{x}, z)$を満たす$z \in \mathbb{N}$があればその最小のzを返し、そうでなければyを返す関数

$$\mu_{z<y}\, p(\vec{x}, z) \stackrel{\text{def}}{=} \mathbf{min}(\{z \in \mathbb{N} \mid z < y, p(\vec{x}, z)\} \cup \{y\})$$

補題 $p(\vec{x}, y)$が原始帰納的述語ならば、$\mu_{z<y}\, p(\vec{x}, z)$ は原始帰納的関数

証明： $(\exists z \leq v)\, p(\vec{x}, z)$の特徴関数 $\prod_{z \leq v} c_p(\vec{x}, z)$を$g(\vec{x}, v)$と書く。

i) $p(\vec{x}, z)$を満たす最小のzが$m\,(< y)$のとき

v	0	1	$\cdots$	$m-1$	m	$m+1$	$\cdots$	$y-1$
$c_p(\vec{x}, v)$	1	1	$\cdots$	1	0	$*$	$\cdots$	$*$
$g(\vec{x}, v)$	1	1	$\cdots$	1	0	0	$\cdots$	0

ここで$*$は0または1

よって、$\sum_{v<y} g(\vec{x}, v) = m$

ii) y未満では$p(\vec{x}, z)$を満たすzがないとき

v	0	1	$\cdots$	$y-1$
$c_p(\vec{x}, v)$	1	1	$\cdots$	1
$g(\vec{x}, v)$	1	1	$\cdots$	1

よって、$\sum_{v<y} g(\vec{x}, v) = y$

例：

- $x \div y$は原始帰納的。なぜなら

  $$x \div y = \mu_{z<x}(x < (y \times (z+1)))$$

- $\mathbf{pr}(x) \stackrel{\mathbf{def}}{=} x$番目の素数

  (例：$\mathbf{pr}(0) = 2$, $\mathbf{pr}(1) = 3$**)**

  は原始帰納的。なぜなら

  $$\mathbf{pr}(0) = 2$$
  $$\mathbf{pr}(x+1) = \mu_{z<\mathbf{pr}(x)!+2}((\mathbf{pr}(x) < z) \wedge \mathbf{prime}(z))$$

帰納的関数

- 原始帰納的でない関数f_Mを計算するチューリング機械Mが存在

  ∵ **(**原始帰納的帰納関数は全域的だが、f_Mは一般には部分関数**)**

- 帰納的関数のクラスは、チューリング機械で計算できる関数のクラスと一致する
- 部分関数fとgは以下を満たすとき等価である**(**$f = g$**)**という
  - 同一の引数を与えたとき、一方が未定義なら他方も未定義であり、そうでなければ結果が一致する

帰納的関数の定義

- 原始帰納的関数
- 合成関数 (帰納的関数の合成)
- 原始帰納法 (帰納的関数を用いた原始帰納法)
- 原始帰納的述語の最小解関数

述語pの最小解関数：

$$\mu_z\, p(\vec{x}, z)$$
$$= \begin{cases} z & \cdots\ p(\vec{x}, y) \text{を満たす} z \in \mathbb{N} \text{が存在するとき} \\ \text{未定義} & \cdots\ \text{それ以外のとき} \end{cases}$$

帰納的関数の例： $f(x) = \mu_z(z^2 = x)$

x	0	1	2	3	4	5	$\cdots$
$f(x)$	0	1	未	未	2	未	$\cdots$

5.3.3 計算モデルの同等性

定理 帰納的関数はチューリング機械で計算できる

証明： 帰納的関数の構成に関する帰納法

- **14**ページの定理**[**原始帰納的関数は**TM**で計算できる**]**
- 原始的述語pの最小解関数$\mu_z\ p(\vec{x}, z)$がチューリング機械で計算できることを示す
- 上の定理の逆も成立する

アッカーマン関数： 帰納的だが原始帰納的でない関数

$$\mathbf{A}(0, y) = y + 1$$
$$\mathbf{A}(x + 1, 0) = \mathbf{A}(x, 1)$$
$$\mathbf{A}(x + 1, y + 1) = \mathbf{A}(x, \mathbf{A}(x + 1, y))$$

問**5.5 (text p.141)** 急激に大きくなる関数で、限定最小解関数では書けない